Revised Due Date: 11:59pm Thursday Nov 15th, 2018.

General Guidelines:
In this assignment, we will implement the front-end of David Lowe's SIFT feature detector [1]. In particular, we will build Gaussian and Difference-of-Gaussian (DoG) scale spaces, and identify keypoints at local DoG scalespace maxima and minima. The scalespace architecture is shown in Fig. 4.11 of the textbook but you can get more detail from the original SIFT paper (provided on Moodle). All of the information you need is in Section 3 of this paper - I recommend that you read this section before starting on the assignment.

On Moodle I have also provided you with two MATLAB functions to help you visualize your results:

showPyr(Pyr, ns) will display a pyramid Pyr (either Gaussian or DoG) with ns subband scales.

showKP(DoGPyr, ns, kp, nkp) will display a DoG pyramid with ns subband scales and plot the most extreme nkp DoG maxima and minima stored in kp. Maxima are shown in red, minima are shown in cyan.

If the pyramids produced by your functions do not work with these visualization routines, your code is not correct.

You are also provided with two simple images that you can use to check that your results make sense.

Submission:
You will submit via Moodle three MATLAB code files. These files must be functions as described below and must be named:

1. GSS.m
2. DoGSS.m
3. SSExtrema.m

Please ensure that you use these exact file names, with correct use of upper and lower case letters, and that your functions exactly satisfy the APIs indicated below. These MATLAB functions will be tested for correct functionality and efficiency. **You do not have to submit a report.** All marks will be awarded based on the functionality and efficiency of your code.

1. (40 marks) **Gaussian Scale Space.**
   Write a MATLAB function GSS that creates a Gaussian pyramid representation of a greyscale image. The API is:

   GPyr = GSS(im, s1, ns, noctaves)

   where

   - **im** is the input greyscale image.
   - **s1** is the space constant of the Gaussian kernel applied to the first (base) level of each octave of the pyramid. Lowe uses s1 = 1.6 pixels.
   - **ns** is the number of subbands per octave. Lowe uses ns = 3.
   - **noctaves** is the number of octaves to represent. Note that each octave is down-sampled by a factor of 2 in both x and y.
   - **GPyr** is a MATLAB cell array containing the Gaussian pyramid. Each element of the array is itself a 3D array containing ns + 3 sub-octave images. This allows computation of ns + 2 DoG images, and thus detection of DoG extrema at ns subband scales.

   Notes:

   - Truncate Gaussian kernels at 1% maximum.
   - Use the 'valid' parameter for conv2 but then pad the image with NaNs to maintain a constant image size within an octave.
   - Useful MATLAB functions: cell, zeros, ceil, sqrt, log, normpdf, sum, length, size, conv2, imresize.

2. (20 marks) **Difference of Gaussian Scale Space.**
   Write a MATLAB function DoGSS that uses GSS to create a Difference of Gaussian pyramid representation of an image. The API is:

   DoGPyr = DoGSS(GPyr)

   where

   GPyr is a Gaussian pyramid (see above).

   DoGPyr is a MATLAB cell array of length noctaves containing the Difference of Gaussian pyramid. Each element of the array is a 3D array containing ns + 2 sub-octave DoG images. This will allow detection of ns extrema of DoG features over scale.

   Useful MATLAB functions: length, cell, diff

3. (40 marks + 10 bonus marks for 4422) **Keypoint Detection.**
   Write a MATLAB function SSExtrema that finds all of the keypoints (scalespace extrema) in a Difference of Gaussian pyramid. The API is:

   kp = SSExtrema(DoGPyr)

   where

   DoGPyr is a Difference of Gaussian pyramid (see above)

   kp is a MATLAB cell array of length noctaves containing the detected keypoints. Each element is itself a cell array over the ns subband scales. Each element of this cell array consists of two fields max and min, each of which is an $n \times 3$ array containing (x, y, val): the coordinates and values of the local maxima and minima within the subband.

   Notes:

   - Local extrema are defined with respect to a local 3 x 3 x 3 neighbourhood of a pixel in scalespace (within an octave).
   - In a compiled language like C or java, these extrema would be found by looping over each pixel in the image, but this is extremely inefficient in MATLAB. An efficient solution can be crafted by using the MATLAB function circshift to stack shifted versions of the image into 27 channels and then detecting the maxima and minima over channels at each pixel. 10 bonus points will be awarded to 4422 students for this efficient solution.
   - Useful MATLAB functions: length, size, find, zeros, cell, circshift, max, min

References

[1] Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (2004), 91–110.