Introduction to Computer Vision (ECSE 415)
Assignment 1: Image Filtering and Edge Detection

Due date: 11:59PM, January 26th , 2024

Please submit your assignment solutions electronically via the myCourses assignment dropbox.

The submission should include a single Jupyter notebook consisting of the solution of the entire assignment. All images and programs should be inserted in the Jupyter notebook, not submitted separately.

Use Python code to implement all operations. You can use OpenCV and Numpy library functions for all parts of the assignment, or you can write your own.

Students are expected to write their own code and assignments. (Academic integrity guidelines can be found at https://www.mcgill.ca/students/srr/ academicrights/integrity).

Assignments received late will be penalized by 10% per day.

Submission Instructions

1. Submit a single jupyter notebook consisting of the solution of the entire assignment.

2. Comment your code appropriately.

3. Do not forget to run Markdown cells (so that the output images appear).

4. Do not submit input/output images separately. Images should be displayed in the jupyter notebook itself. Assume input images are kept in the same directory as the codes.

5. Make sure that the submitted code is running without error. Describe any specific requirements for running the code in your notebook.

6. If external libraries were used in your code please specify their names and versions in the notebook.

7. We are expecting you to make a path variable at the beginning of your codebase.
This should point to your working local (or google drive) folder.
Your path variable should be defined at the top of your Jupyter notebook.
While grading, we are expecting that we just have to change the path variable once and it should allow us to run your solution smoothly on our own system.

1 Image Acquisition (5 Points)

Using a cellphone camera or a standalone digital camera, capture ***two images*** of a household object, where each image of the object is ***taken from a different viewpoint***. These images can be of any size. An example of such an image pair is shown below:

Please note that all assignments are to be done individually (not in groups), and so you must acquire your own images. Do not share images or copy someone else's images!

You will be using these images, as well as the processed images, in some future assignments, so it is important to do all the steps correctly. Display the original images in the assignment's Jupyter notebook.

2 Convert to Grayscale (5 Points)

If your images are color (RGB), convert them to Grayscale, by averaging each pixel's R, G, and B values. Display the grayscale images in the assignment's Jupyter notebook.

3 Smooth the images using Gaussian smoothing (10 Points)

Smooth the pair of grayscale images using a 5x5 pixel Gaussian kernel. Then repeat the smoothing on the original grayscale images, this time using a 11x11 Gaussian kernel. Display the smoothed images in the notebook.

4 Compute Image Gradients (10 Points)

Compute the x and y derivative images of the smoothed images using the horizontal and vertical Sobel filters. Display the derivative images in the notebook.

5 Compute the Edge Magnitude and Orientation (10 Points)

Compute the edge gradient magnitude and orientation of the smoothed images using the Sobel filter values. Display the magnitude and orientation images in the notebook. For the orientation image, display the angle value using an RGB colormap, such as 'jet' in the imshow() function (e.g. something like ax.imshow(data, cmap='jet')).

Details about matplotlib colormaps, including other colormaps you can try, can be found at: https://matplotlib.org/stable/tutorials/colors/colormaps.html

6 Canny Edge Detection with opencv (10 Points)

Setup opencv in your colab or home computer environment (go to the Tutorial to learn how to do this). Use the Canny edge detector implementation in opencv to compute the Canny edge detector on your smoothed images. (look at https://docs.opencv.org/3.4/da/d22/tutorial_py_canny.html for details).

Plot the Canny detector output images in the notebook.