Intro to Image Understanding (CSC420)

Available: October 17^{th}, 2020
Due Date: October 30^{th}, 10:59 pm, 2020
Total: 150 marks

Notes: You are encouraged to become familiar with the **Python** environments to use it for the experimental parts of the assignment. We EXPECT EVERYONE TO SUBMIT **ORIGINAL WORK** FOR ASSIGNMENTS. This means that if you have consulted anyone or any sources (including source code), you must disclose this explicitly. Anything you submit must reflect your work.

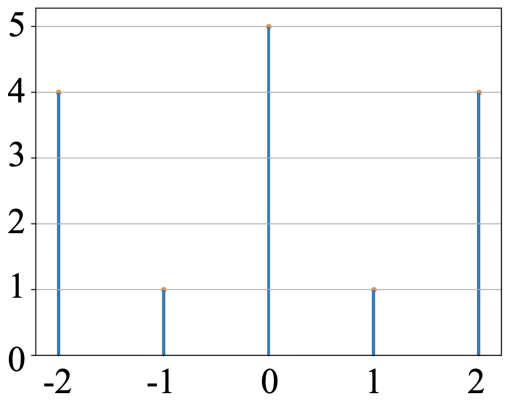
Submission Instructions:

- Your submission should be in the form of an electronic report (PDF), with the answers to the specific questions (each question separately), and a presentation and discussion of your results. For this, please submit a file called "**report.pdf**" to MarkUs directly.
- Submit documented codes that you have written to generate your results separately. Please store all of those files in a folder called Assignment2, zip the folder, and then submit the file Assignment2.zip to MarkUs. You should include a **"README.txt"** file (inside the Assignment 2 folder) which details how to run the submitted codes.
- Don't worry if you realize you made a mistake after submitting your zip file: you can submit multiple times on MarkUs.

Part I: Theory (40 marks)

Q1) (10 marks)

Upsample by a factor of 4 the discrete-time signal shown in figure below using linear interpolation. Show your work and plot your result.



Q2) (15 marks)
Consider a signal with its analytical form given by equation:

$$x(t) = \sin^2(2F_a\pi t) + \frac{1}{7}\cos(F_b\pi t) + \frac{1}{5}\sin(5F_c\pi t)$$

Using the Nyquist–Shannon sampling theorem, what is the sufficient sample-rate for this signal (based on constant numbers F_a, F_b and F_c) so it could be possible to recover the original signal from the discrete samples?

Q3) (15 marks)
For corner detection, we defined the Second Moment Matrix as follows:

$$M = \sum_x \sum_y w(x,y) \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$

Let's call the 2x2 matrix used here N, i.e.:

$$N = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$

1. **(5 marks)** In class (Lecture 5 - Part 1) we showed that one of the eigenvalues of N is zero ($\lambda_1 = 0$). What is the value of its other eigenvalue? i.e. $\lambda_2 = ?$

2. **(10 marks)**
   Assuming that matrix N is positive semi-definite*, show that matrix M is positive semi-definite.

   * see lecture 5 scribbles for the definition of positive definite matrices

Part II: Image Resizing with Seam Carving (75 marks)

Text is borrowed from: `https://en.wikipedia.org/wiki/Seam_carving`.

Seam carving (or liquid rescaling) is an algorithm for content-aware image resizing, developed by Shai Avidan, of Mitsubishi Electric Research Laboratories (MERL), and Ariel Shamir, of the Interdisciplinary Center and MERL. It functions by establishing several seams (paths of least importance) in an image and automatically removes seams to reduce the image size or inserts seams to extend it. Seam carving also allows manually defining areas in which pixels may not be modified and features the ability to remove whole objects from photographs.

The purpose of the algorithm is image retargeting, which is the problem of displaying images without distortion on media of various sizes (cell phones, projection screens) using document standards, like HTML, that already support dynamic changes in page layout and text but not images.
Imagine we have an image like the one shown in Figure 1a and want to retarget that for a specific size. Here in this example, the original image shown is of size 968×1428 and the

a b c d

Figure 1: a) Original image. b) Scaled image. c) Cropped image. d) Seam carved image.

target size is 968×957. There are multiple strategies to achieve this target output. One is the scaling image just in the horizontal direction (subsampling like as we discussed in the class). The result would look like the one shown in Figure 1b which is undesirable because the castle is distorted. The other idea is to crop the image as shown in Figure 1c, again, Cropping is undesirable as well because part of the castle is removed. The one shown in Figure 1d is obtained from the Seam Carving algorithm which allows for effective resizing of images. This algorithm allows the image to be resized without losing meaningful content from cropping or scaling. In this part, you are required to implement Seam Carving from scratch and test it on different images.

The algorithm steps include:

1. Compute magnitude of gradients of an image like how you did for assignment 1:

$$g(x, y) = |\nabla f(x, y)| = \sqrt{g_x^2 + g_y^2}$$

2. **(45 marks)** Find the connected path of pixels that has the smallest sum of gradients. A path is valid if it is connected (the neighboring points in the path are also neighboring pixels in the image), it starts in the first row of the image and in each step continues one row down. It finishes in the last row of the image.

   To implement this part, you can use "dynamic programming" method. Assume the energy function you are trying to minimize is represented by $f(r, c)$ where $f(r, c)$ is the smallest sum of gradients on the connected path that starts from a pixel at row r and column c and goes down row by row until it finishes in the last row of the image. First of all, propose a recurrence relation equation based on $f(r, c)$ and then implement that to find the connected path of pixels with the smallest sum of gradients.

3. **(10 marks)** At each iteration, remove the pixels in the path from the image. This gives you a new image with one column less. Remove paths with the lowest sum of gradients until you achieve the desired dimension.

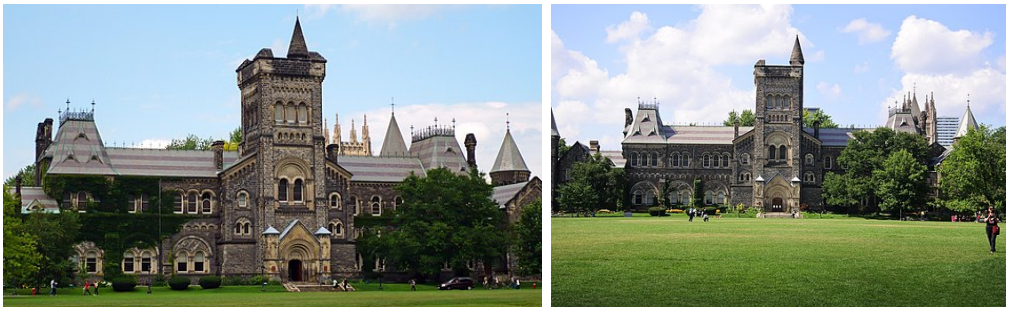
4. **(10 marks)** To be able to remove rows, you can first transpose your image matrix then remove the desired number of columns using the steps detailed above, and finally transpose the outcome to get the outcome image with less number of rows.

(10 marks) The input to the algorithm above includes an image and the desired size (which is smaller than the original image size). When you have your algorithm implemented test on the given images on Quercus with their associated desired sizes. Compare your results with cropping and scaling functions (you can use built-in libraries to crop and scale).

Part III: Corner Detection (35 marks)

Download two images (I_1 and I_2) taken from the same building but from two different viewpoints, e.g. two images of the UofT University College:

1. `https://commons.wikimedia.org/wiki/File:University_College,_University_of_Toronto.jpg`
2. `https://commons.wikimedia.org/wiki/File:University_College_Lawn,_University_of_Toronto,_Canada.jpg`



1. Calculate the eigenvalues of the Sencond Momend Matrix (M) for each pixel of I_1 and I_2.
2. **(10 marks)** Show the scatterplot of λ_1 and λ_2 for I_1 and the same scatterplot for I_2.
3. **(10 marks)** Based on the scatterplots, pick a threshold for $min(\lambda_1, \lambda_2)$ to detect corners. Illustrate detected corners on each image using the chosen threshold.
4. **(15 marks)** Constructing matrix M involves the choice of a window function, $w(x, y)$. Often a Gaussian kernel is used. Repeats steps 1, 2, and 2 above, using a (significantly) different Gaussian kernel (i.e. a different σ) than the value you used before. For example, choose a σ that is significantly larger than the previous one. Explain how this choice influenced detected corners in each image.