**Homework Assignment 1**
**CSCI E-25 Image Processing and Computer Vision**
**Due Date: October 2, 2019**

**Problem 1 [10 points]**
Camera calibration using vanishing points. Consider the image below of size 470x710:

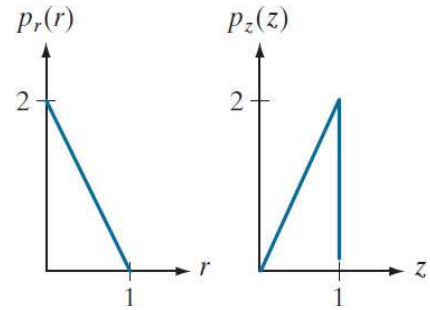
(a) Using the highlighted lines, calculate the three vanishing points, $v_1, v_2, v_3$, associated with each colored line pair. Notice that these three line pairs are associated with mutually orthogonal planes. Approximate (x,y) pixels coordinates for the start and end of these lines are:
Yellow: L1 (312,181)<->(4,289), L2 (312,316) <-> (7,350)
Red: L1 (424,195)<->(597,233), L2 (430,341) <-> (601,344)
Green: L1 (60,434)<->(627,383), L2 (323,447) <-> (673,390)
*Hint: These course notes contain some useful material:*
http://web.stanford.edu/class/cs231a/course_notes/02-single-view-metrology.pdf
*You might find the cross-product useful for finding intersection points:*
*https://en.wikipedia.org/wiki/Line%E2%80%93line_intersection#Using_homogeneous_coordinates*
Find the vanishing points in homogeneous coordinates assuming that the line normal vectors are normalized to unit norm, and 2D pixel coordinates.

(b) Using the assumption that the camera has zero skew and square pixels, write the form of the matrix $\omega = (KK^T)^{-1}$, where K is the camera calibration matrix that depends on $\alpha, c_x, c_y$.

(c) Write the vector of nonzero elements of $\omega$ as *w* (hint: should have length 4), and write the three constraints as a linear system $Aw = 0$, where $A$ is a 3x4 matrix, taking into account the fact that the vanishing directions are mutually orthogonal. Find the matrix $\omega$. (hint: think of null vector of $A$ or use SVD). Show your work.

(d) Find the calibration matrix K using matrix inversion and Cholesky factorization.

**Problem 2 (DIP-4e 3.14) [5 points]**
An image with intensities in the range [0,1] has the probability density function $p_r(r)$ shown in the figure below. The goal is to transform the intensity levels to match the specified probability density function $p_z(z)$ shown below. Assuming continuous quantities, find the transformation (express in terms of r and z) that will accomplish this.




**Problem 3 (DIP-4e 3.26) [5 points]**
Consider the following kernel, w, and image, f:

$$w = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, f = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) Give a sketch of the area performing the convolution operation centered at point (2,3) ($2^{nd}$ row, $3^{rd}$ column) of the image shown above. Show specific values of w and f.
(b) Compute the convolution using the minimum zero padding needed. Show the details of your computations when the kernel is centered at (2,3) on f, and then show the final full convolution result.

**Problem 4 [15 points]**
K-means clustering. In this problem, you will be deriving the equations for the K-means algorithm and using it to cluster some two-dimensional Gaussian data. Let $\mu_k$ denote the d-dimensional prototype vectors associated with the k-th cluster (e.g., cluster center). The goal is to find an assignment of all data points to clusters, as well as a set of prototype vectors such that the sum of squares of distances of each data point to its closest prototype vector is minimized. Let $r_{nk}$ denote the binary variable indicating if data point $x_n$ is assigned to cluster k. The distortion measure to minimize is:

$$J = \sum_n \sum_k r_{nk} \parallel x_n - \mu_k \parallel_2^2$$

(a) As the first stage to the iterative procedure, assume that the cluster means $\mu_k$ are fixed. Find the expression for the $\{r_{nk}\}$ variables that minimize J.
(b) The second stage of the iterative procedure assumes that the assignment variables $r_{nk}$ are held fixed. Find the expression for the cluster means $\{\mu_k\}$ that minimize J.
(c) Write a program in Python that takes as inputs the data and an initialization, and runs the two steps (a), (b) iteratively until convergence.
(d) Apply the program from part (c) on the Gaussian dataset "gaussian2D.npy" composed of K=3 clusters, and display the clustering progress as a function of iteration (by computing J), and show the final result visually in 2D. In addition, please output the estimated cluster means.

**Problem 5 [15 points]**
Histogram equalization. You may use OpenCV.

(a) The image "boston0.png" is a low-contrast grayscale image. Implement the histogram equalization algorithm on this image using the CDF method described in class. Explain what this does to the image.
(b) Compare the result you obtained from part (a) with the OpenCV histogram equalization method **cv2.equalizeHist()**.

(c) Do you think a histogram specification approach might work better? If yes, please implement an appropriate histogram specification method, and explain any visual differences from the result (a). If not, explain why not.

(d) Repeat the steps (a)-(c) for the image "budapest0.png". Make sure to explain your reasoning. The end goal is to obtain the most visually appealing result by enhancing the image features.

**Problem 6 [5 points]**

Lowpass filtering.

(a) Consider the image "skier0.jpg". Implement lowpass filtering using a 5x5 averaging kernel, and a 5x5 Gaussian kernel using spatial convolution with $\sigma = 1.0,2.0,3.0$. Explain what this does to the image.

(b) Compare the result you obtained in part (a) with the OpenCV filtering method **cv2.filter2D()**.

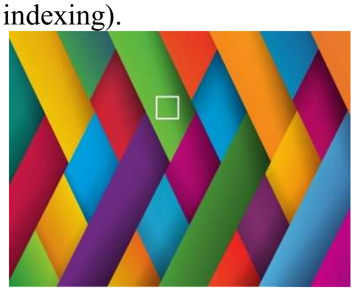
(c) How many multiply-and-add operations are needed to implement this spatial convolution? (Big-O notation is fine here)

(d) Are these kernels separable? Explain. If yes, how would the computations in part (c) be minimized given that the kernels are separable?

**Problem 7 (Graduate Credit) [10 points]**

Color-based segmentation in RGB color space.

(a) Consider the image "multicolor-design.jpg", and the highlighted region enclosed by corner coordinates $(x_1, y_1) = (270,120)$ and $(x_2, y_2) = (310,160)$ (assuming OpenCV coordinate indexing).



Compute the mean, $\mu = (\mu_R, \mu_G, \mu_B)$, and standard deviations, $\sigma = (\sigma_R, \sigma_G, \sigma_B)$, of the RGB pixels in the highlighted region.

(b) Form two types of regions in RGB space centered at $\mu$. First, a Euclidean ball $D(z, \mu) = \|z - \mu\|_2 \leq D_0$ and second, an $\ell_\infty$ ball (box) $D(z, \mu) = \| z - \mu \|_\infty \leq D_0$, for various $D_0$. Segment the image by setting RGB points within the designed region to white and the remaining points to black. Your goal is to segment the green stripes in the image as well as possible. Which norm/metric is better after tuning? Why?

(c) Repeat the segmentation experiment from part (b) with the following scaled metric:

$$\| K_\sigma^{-1}(z - \mu) \|_\infty \leq c$$

where $K_\sigma = diag(\sigma_R, \sigma_G, \sigma_B)$, and $c$ is a value slightly larger than one, e.g., $c = 1.5$. Experiment with the best value for $c$. Does this metric yield better segmentation results than the ones in part (b)? Why or why not?

**Undergraduate Total Points: 55**

**Graduate Total Points: 65**