CS 6643 **E. K. Wong**
Project 2: Human Detection using HOG Feature **Spring 2023**

Total # of points = 100.

Project description. Write a program to compute the *HOG* (*Histograms of Oriented Gradients*) feature from an input image and then classify the HOG feature vector into *human* or *no-human* by using a *3-nearest neighbor* (*NN*) *classifier*. In the 3-NN classifier, the distance between the input image and each of the images in a sample database is computed and the classification of the input is taken to be the majority classification of the three nearest neighbors. In computing the distance, use the histogram intersection and Hellinger distance metrics (separately) below and report the classification accuracy for each. In the distance metrics below, I is the *normalized* HOG feature vector for the input image and S is the *normalized* HOG feature vector of an image from the sample database. The subscript j indicates the jth component and n is the dimension of the feature vector. (See below for the computation of normalized HOG feature.)

- Histogram intersection

$$1 - \sum_{j=1}^{n} \min(I_j, S_j)$$

- Hellinger distance

$$1 - \sum_{j=1}^{n} \sqrt{I_j S_j}$$

Conversion to grayscale: The input images and sample database images are color images cut out from larger images. First, convert the color images into grayscale using the formula $I = Round(0.299R + 0.587G + 0.114B)$ where R, G and B are the pixel values from the red, green and blue channels of the color images, respectively, and *Round* is the round off operator.

Gradient operator: Use the Sobel's operator for the computation of horizontal and vertical gradients. Use the formula $M(i,j) = \sqrt{G_x^2 + G_y^2}$ to compute gradient magnitude, where G_x and G_y are the horizontal and vertical gradients. Normalize and round off the gradient magnitude to integers within the range [0, 255]. Next, compute the gradient angle. For image locations where the templates go outside of the borders of the image, assign a value of 0 to both the gradient magnitude and gradient angle. Also, if both G_x and G_y are 0, assign a value of 0 to both gradient magnitude and gradient angle.

Normalized HOG feature: The *normalized* HOG feature vector can be obtained by dividing each component of the HOG feature by the sum of all the components. Refer to the lecture slides for the computation of the HOG feature. Use the unsigned representation and quantize the gradient angle into one of the 9 bins as shown in the table below. If the gradient angle is within the range [180, 360), simply subtract the angle by 180 first. Use the following parameter values in your implementation: *cell size* = 8 x 8 pixels, *block size* = 16 x 16 pixels (or 2 x 2 cells), *block overlap* or *step size* = 8 pixels (or 1 cell.) Use *L2* norm for block normalization. Leave the histogram and final feature values as floating point numbers (Do not round off to integers.)

Histogram Bins		
Bin #	Angle in degrees	Bin center
1	[0,20)	10
2	[20,40)	30
3	[40,60)	50
4	[60,80)	70
5	[80,100)	90
6	[100,120)	110
7	[120,140)	130
8	[140,160)	150
9	[160,180)	170

Sample database and test images: A sample database and a set of 10 test images in *.bmp* format will be provided on Brightspace for you to test your program. The sample database and the test images contain both positive (human) and negative (no human) examples. The images are of size 160 (height) X 96 (width). For the given image size and the parameters given above, there will be 20 X 12 cells and 19 X 11 blocks in the detection window, and the dimension of the resulting HOG feature is 7,524.

Implementation: You can use Python, C++/C, Java or Matlab to implement your program. If you would like to use a different language, send me an email first. You are not allowed to use any built-in library functions for any of the tasks you are required to implement, including the Sobel's operator and the computation of the HOG features and the two distance metrics. The only library functions you are allowed to use are those for the reading and writing of image files, matrix and vector arithmetic, and other commonly used mathematical functions.

Hand-in: Hand in the following files on BrightSpace by the due date. Please submit as separate files, do not put into a ZIP file.

- Your source code file. Put comments in your source code to make it easier for someone else to read your program. Points will be taken off if you do not have comments in your source code.
- The ASCII (.txt) files containing the HOG feature values for three of the sample images from the database and three of the test images (one file per image.) I will let you know which images to provide the HOG feature values later. The feature values should be separated by line breaks. You should have 7,524 lines in each file.
- A PDF report that contains the following:
    - Instruction on how to run your program and instruction on how to compile your program if your program requires compilation.
    - Normalized gradient magnitude images for the 10 test images.
    - The source code of your program (Copy-and-paste from source code file. This is in addition to the source code file that you need to hand in.)
    - Use the table below to report the classification results for each of the distance metrics above. Use a separate table for each metric.

Test input image	**Correct Classification**	**File name of 1st NN, distance & classification**	**File name of 2nd NN, distance & classification**	**File name of 3rd NN, distance & classification**	**Classification from 3-NN**
Image 1	Human				
Image 2	Human				
Image 3	Human				
Image 4	Human				
Image 5	Human				
Image 6	No-human				
Image 7	No-human				
Image 8	No-human				
Image 9	No-human				
Image 10	No-human				