Comp. 4102: Assignment #2
Due: Tuesday Feb. 28, 2017 at 4:00 PM
Submit this assignment using CuLearn
Tell me which version of OpenCV that you used

1) The goal of the program you are going to write is to classify street signs, in particular stop signs and speed limit signs. There are four images that you will use to test your program; speedsign3.jpg – a 40km speed sign, speedsign12.jpg – an 80km speed sign, speedsign4.jpg – a 100km speed sign and finally stop4.jpg – a stop sign. I have given you a program called classify_sign_template.cpp, which you should change to do the classification. The result will be a copy of the input image with the appropriate label (a stop sign, speed limit 40, speed limit 80, or no match) for each of these three input images. There are two parts to the program, the first decides on which type of sign you are looking at (stop sign or speed limit). To do this you will use the routines Canny, findContours, and approxPolyDP. To help you I am giving you some information on the parameters of these routines. First of all, `Canny(, , canny_thresh, canny_thresh*2, 3)` where canny_thresh is defined in the program I have given to you. Also, `findContours(, , , RETR_EXTERNAL, CHAIN_APPROX_NONE, Point(0, 0))`, and `approxPolyDP(, , contours[i].size()*.02, true)`. These are the parameter values which worked for me, you can use them if you want, but you do not need to use them. The second part of the program will use the routines `getPerspectiveTransform`, and `warpPerspective` along with images speed_40.bmp, and speed_80.bmp to tell the type of speed limit sign, but only once you have determined that your are indeed looking at a speed limit sign. Hand in the source code, and tell me the version of OpenCV that you are using. You should also include the four output files that you have produced, one for each of the four test cases. **7 marks**

2) Use the program called harris_corner_template.c and add some code to find the corners in the image checkers.jpg. You should say that a pixel in the image is a corner if it passes the given threshold for Harris Corners. You do not need to thin the corners using non-maxima suppression. You should draw a small circle or dot on these corner pixels, and then save the resulting image. The final image should look similar to Corner-out.pgn. You need to look up the formula to find the eigenvalues of a real, symmetric, two by two matrix. This is described in http://people.math.gatech.edu/~klounici6/2605/Lectures%20notes%20Carlen/chap3.pdf You should submit your source code along with the final output image. Do not use the routines cornerEigenValsAndVecs, cornerHarris, preCornerDetect, cornerMinEigenVal in this question. I want you to compute the required quantities in the long way, without using these routines as shortcuts. **3 marks**