# **CVI620/DPS920- Lab 7**

# **Edges & Lines**

| Total Mark: | 10 marks (2.5% of the total course grade)   * 7 out of 10: Blackboard submission * 3 out of 10: Lab demo |
| --- | --- |
| Submission file(s): | * Lab07\_1.cpp * Lab07\_2.cpp * Lab07\_20W.docx including the **result images and answers** |

Please work in **groups** to complete this lab. This lab is worth 2.5% of the total course grade and will be evaluated through your written submission, as well as the lab demo. During the lab demo, group members are *randomly* selected to explain the submitted solution. Group members not present during the lab demo will lose the demo mark.

1. Add this declaration to your file:

We, ------------ (mention your names), declare that the attached assignment is our own work in accordance with the Seneca Academic Policy. We have not copied any part of this assignment, manually or electronically, from any other source including web sites, unless specified as references. We have not distributed our work to other students.

**Whenever you are asked to display an image in the lab instructions, please paste the resulting images in this document.**

## **Part I: Edge Detection**

1. Write code in Lab07\_1.cpp to
   1. Open an image and convert it to greyscale. Display the greyscale image.
   2. Find the vertical edges using filter2D and the 3x3 vertical kernel. Display results.
   3. Find the horizontal edges using filter2D and the 3x3 horizontal kernel. Display results.
   4. Calculate the magnitude of edges at each pixel using the vertical and horizontal edge maps. Display the magnitude edge map.
   5. Use the OpenCV Sobel function to find the edges on the image. Use a 3x3 Sobel filter. Display the results and compare with results in (d). Are they the same? If not, why do you think they look different?
   6. Compare the edge map in (e) with the result of applying a Laplacian 3x3 filter. Display the result.
   7. Compare your result for (f) with the result of applying a Canny edge detector with the default parameters. Experiment with different threshold values and display the results. How is the Canny map different from the previous maps?

## **Part II: Line Detection**

1. Create a program (save as Lab07\_2.cpp). Include code to:
   1. Open the Yonge Street image and convert to greyscale by changing the flag in *imread.* Display the greyscale image.
   2. Use the Canny edge detector for the next steps. Which threshold values did you use? Display the edge map.
   3. Use the Hough transform to detect lines in the image and display on the image. Increase the threshold enough to detect only two lines in the image. What threshold value did you use?
   4. Find the vanishing point by finding the intersection of the above lines:

Intersection point (ri, ci):

* 1. Use the Hough transform to detect the circles in the image by tuning the parameter. Is it easy to find the circles in the image? Why?
  2. Show the image with the line segments and circles drawn on the image.

1. Specify what each member has done towards the completion of this work:

|  | Name | Task(s) |
| --- | --- | --- |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |

[Yonge Street image link: https://upload.wikimedia.org/wikipedia/commons/thumb/e/e3/Yonge\_Street\_south\_of\_College\_St.%2C\_Toronto.JPG/512px-Yonge\_Street\_south\_of\_College\_St.%2C\_Toronto.JPG]