INTRODUCTION TO DIGITAL IMAGE PROCESSING

ASSIGNMENT 4

Due date: Thursday, November 16, 2017 by 5 pm
Total marks: 5
Late penalty: 1 mark per day overdue. Late assignments will not be accepted after 5 pm on Monday, November 20, 2017, and a mark of zero will be given.

All assignments will be done in **groups** of 3-4, and the same final mark for the assignment will be assigned to all group members. For convenience, we prefer if you keep the same group as for previous assignments. However, if you wish to change groups, please send Prof. Ladak an e-mail (hladak@uwo.ca) indicating which group you are currently part of and which group you wish to change to. Instructions for checking your current group membership are given in the Appendix.

Instructions for submitting answers are given with each question below. Once one member submits an assignment, it is closed for resubmission by the group, so all group members should agree to submit before actually submitting online.

CONVENTIONS

Fixed-point font (`Courier`) is used to denote MATLAB commands, variables and filenames.

OBJECTIVES

The main objective is to give you exposure to a real-world application that can potentially be solved using what you have learned in the course thus far. A secondary objective is to give you experience using colour images.

PRODUCTION LINE VISUAL INSPECTION: SOFT-DRINK BOTTLING PLANT

The most common use of image processing in an industrial setting is for the automated visual inspection of products leaving a production facility. Automated inspection is used to inspect everything from pharmaceutical drugs to textile production. It is estimated that the majority of products bought on supermarket shelves are inspected using automated "machine vision" based systems prior to dispatch. Why? - to avoid the cost of shipping a faulty or sub-standard item to a supermarket shelf that no-one wants to buy!

In this practical exercise we are dealing with a bottling production line in a facility bottling Coca-Cola for the domestic market. We have a set of images of the bottles as they leave the bottling line; these are taken under near constant lighting conditions. The bottling company requires a vision system to automatically identify a number of different faults that may occur during filling, labelling and capping stages of production so that these bottles can be intercepted prior to packaging and shipping.

Your task is to design and prototype an image processing system to detect the set of fault conditions that may occur together with identifying the type of fault that has occurred. You will develop this prototype system using MATLAB. The following faults may occur in a bottling plant:

1. bottle cap is missing
2. bottle over-filled
3. bottle under-filled
4. bottle has label missing
5. bottle has label but label printing has failed (i.e. label is white)
6. bottle label is not straight
7. bottle is deformed (i.e. squashed) in some way.

ASSIGNMENT PROBLEMS

For this assignment, you are required to develop a visual inspection system that correctly identifies the first 3 faults only:

1. bottle cap is missing
2. bottle over-filled
3. bottle under-filled.

Typical images acquired by the high-speed bottle inspection camera are shown in Figure 1.

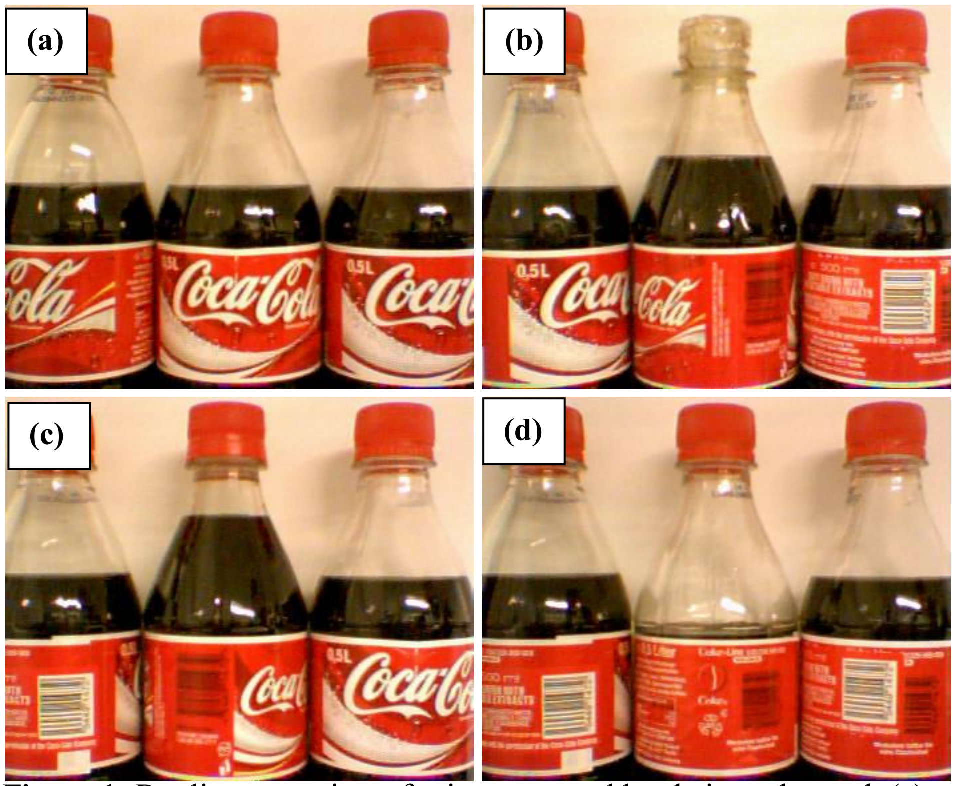

Figure 1: Bottling scenarios referring to <u>central</u> bottle in each panel. (a) Desired – bottle is capped and filled to correct level, (b) not capped, (c) overfilled, and (d) underfilled.

In each image we are only interested in classifying the <u>central</u> bottle in the image. One image is taken for each bottle leaving the production line so faults occurring in bottles at the sides will be detected separately when these particular bottles are themselves photographed central to the image. Faults with side bottles must be ignored by your system, and only the three faults above must be reported.

The instructor has implemented code to get you started. First, download the file ‘test.zip’ from OWL and decompress it in your MATLAB working directory; this should generate a sub-directory called ‘test’ in your MATLAB working directory. Download the M-file called ‘check_bottle.m’ into the ‘test’ directory. Run the code. It correctly detects missing caps, but does not detect whether a bottle is overfilled or underfilled; it simply (and sometimes incorrectly) states that each bottle is normally filled. **Your task is to understand the code and then write additional parts where indicated in the M-file to test for overfilling and underfilling.** Note that as this is only a prototype – efficiency of the approach is less important than accuracy; however, the simplicity of the selected image-processing approach lends itself well to high-speed implementation. Please test your code on all of the test images that have been provided.

All code and answers requested below must be submitted using OWL. To provide answers via OWL:

1. **One group member should log into OWL and select "ECE 4445A 001 FW17".**
2. **From the left-hand side, select "Assignments".**
3. **From the page that comes up, select "Assignment 4".**
4. **You will now reach the submission page for Assignment 4. Follow the instructions below for each part to submit answers.**

(a) **[4 marks]** After you have added code for checking the overfilling and underfilling conditions and tested it on the test images, save the script in a file called `check_bottle_#.m` where # is your group number. For instance, if you are part of group 1, your filename would be `check_bottle_1.m`. If you are part of group 50, your filename would be `check_bottle_50.m`. **NOTE: Use the exact filename and function name as specified here. All letters are in lowercase. Your function should be commented.**

When you are on the submission page in OWL for Assignment 1, scroll to the bottom and attach your M-file. Also, cut and paste this code into the text box taking care to label this as part (a).

Answer the following questions on the submission page of OWL in the text box, taking care to include labels such as (b) and (c).

(b) **[0.5 marks]** Manufacturing processes are tightly controlled and maintained to regulate factors such as object (bottle) position relative to the camera and to regulate lighting conditions. Could possible shifts in bottle position (either up or down relative to the camera) affect the results? Explain why or why not bottle position may or may not affect fault detection.

(c) **[0.5 marks]** Image processing algorithms often have a number of parameters that are usually set empirically, such as the positions of the various rectangles and thresholds used in the proposed partial solution. Is your final code sensitive to the location of `under_rect` and to the threshold `under_thres`? Explain why or why not. In answering this question, you may wish to consider running your code with the following values:

```
under_x = [165, 185, 185, 165];
under_y = [150, 150, 165, 165];
under_thres = 0.3
```

Again, each group should only make one submission. In the text box titled on the submission page, enter the name and student number of each group member.

RESOURCES

1. A good description of the RGB colour model can be found at: https://en.wikipedia.org/wiki/RGB_color_model
2. A reasonable description of the HSV colour model can be found at: https://en.wikipedia.org/wiki/HSL_and_HSV
3. All MATLAB guides can be found at:
   http://www.mathworks.com/access/helpdesk/help/helpdesk.shtml
   For information on the image processing toolbox, select the link labelled "Image Processing Toolbox".

APPENDIX

1. Log onto OWL and click on Site Info on the left-hand side when you are on the course site.

2. Click Groups you are a member of**.** The Assignment Group you are a member will come up.