Machine Perception
COMP3007

ASSIGNMENT

Due Date: Week 11 - Friday 11 October 2019 at 5pm.
Weight: **30%** of the unit mark.

Note: *This document is subject to corrections and updates. Announcements will be made promptly on Blackboard and during lectures. Always check for the latest version of the assignment. Failure to do so may result in you not completing the tasks according to the specifications.*

Your total score for this assignment will need to be at least **20 marks** out of the total 100 marks. You will fail this unit if you cannot meet this basic pass mark, regardless of your scores in the mid-semester test and final exam.

1 Overview

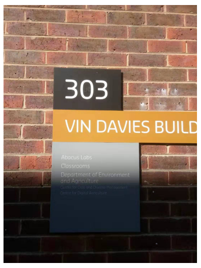
This assignment provides an opportunity for you to demonstrate how you can use what you have learned from lectures and practicals to develop computer vision algorithms for a real world application problem. For a successful completion of this assignment, you need to understand the fundamental algorithms covered in the lectures, conduct some research into the machine perception problem to design suitable features and classification methods, and use the skills that you have developed through completing practical exercises to build essential components of a computer vision algorithm. External codes except for OpenCV are NOT allowed to use in your assignment. Feel free to use the work you have done in your practical exercises.

A substantial attempt for this assignment is required to pass this unit. A mark of 20$\%$ or more is considered a substantial attempt. This means you will not pass this unit if your total mark of this assignment is lower than 20%, even if you achieve full marks in your mid-semester test and final exam.

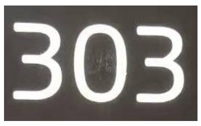
2 Background

In this assignment, you will develop two computer vision programs to detect and recognise the building signages (Figure 1) and directional signages (Figure 2) from images taken in the Bentley Campus of Curtin University. A typical machine perception approach to solving this problem would consist of at least the following

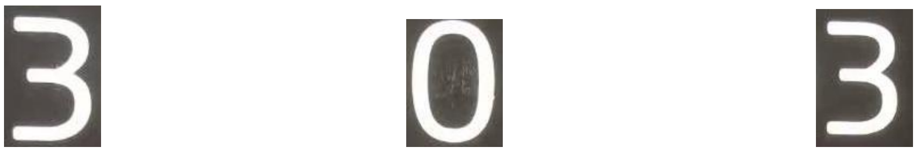
- Read an input image;
- Perform necessary image-processing operations;
- Detect and localise the signage;



(a) Input Image



(b) Detected Area



(c) Extracted Digits

Figure 1: Building signage detection and recognition.

- Segment the numbers and direction signs into individual digits and individual directional signs;
- Recognize individual digits and directional signs.

You will implement a suitable machine perception pipeline to perform the above steps by writing Java/Python programs. Your implementation is primarily evaluated in terms of the two following tasks:

- Task 1: Building signage detection and recognition;
- Task 2: Directional signage detection and extraction.

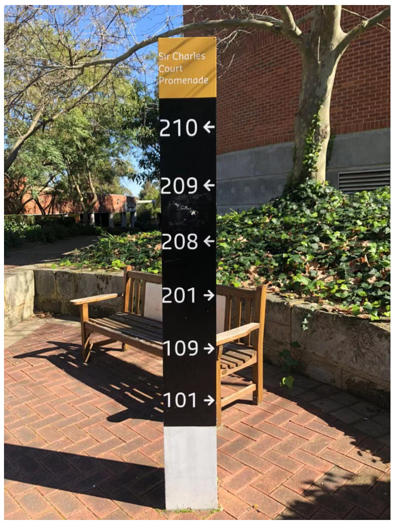
It is expected that you will make use of the computer vision algorithms discussed in the lectures and the skills acquired through doing practical exercises in order to complete the required tasks. You will also need to conduct your own research in order to understand different approaches to solving each task and decide your own choice of implementation and/or invention according to the specific settings of this assignment.

3 The Tasks

3.1 Task 1: Building signage detection and recognition

Develop a program that reads in colour images from a specified directory. For each image, detect the digits, extract and classify the digits, and finally output the building number, which typically includes three digits (e.g 303) in the Bentley campus of Curtin University. Your program is considered working if

- The detected area meets the following criteria
  - It must be a rectangle shape not exceeding the maximum allowable size, which is specified as follows
    - ∗ The overlap of the detected area with the ground truth must contain at least 50% of the detected area and contain at least 50% of the ground truth area.



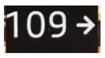
(a) Input Image

(b) Building #1

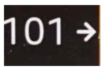
(c) Building #2

(d) Building #3

(e) Building #4



(f) Building #5



(g) Building #6

Figure 2: Directional signage detection and building number extraction.

  - It must contain all the digits of the building number.

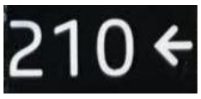
- It must extract all the three digits;
- It must recognize the three digits correctly

An illustrative example for building number detection and recognition is provided in Figure 1. A data set of digits will be provided for you to train the digit classifiers.

Note that in this task, each test image will contain only one building number. Thus, your program *must not report more than one detected area*. Otherwise, it will be considered a failed detection.

3.2 Task 2: Directional signage detection and recognition

Write another program that reads in colour images from a specified directory for directional signage images. For each image, detect all the building numbers and directional signs and classify them to output a list of building numbers along with their directional signs (e.g. turn left). This task consists of three major steps. In the first step, you need to develop algorithms which can detect and extract all the building numbers followed by a directional sign. An example of an input image and outputs is demonstrated in Figure 2. In the second step, you need to extract the individual digits and individual directional signs, as demonstrated in Figure 3. In the third step, you need to train a classifier to classify the digits and directional signs, and output a list of all the building numbers along with their directional signs. A data set of digits and directional signs will be provided for you to train the classifier.


(a) Extracted building number followed by a directional sign.

(b) Digit #1

(c) Digit #2

(d) Digit #3

(e) Directional sign

Figure 3: Building number and directional sign recognition.

4 Specifications and Marking Guide

4.1 Report: 50 marks (Task 1: 20 marks; Task 2: 30 marks)

A written report must be submitted, in PDF format, to Blackboard by the due date. This submission must contain

- A completed assignment cover sheet
- Printout of your source code
- A document that includes:
  - Statements on how much you have attempted the assignment.
  - The detail of your implementation for each task: this must clearly indicate your approach, the features you extract, the methods you use for detection, segmentation, and digit recognition, etc. It must allow the marker to understand how you approach the machine perception tasks.
  - The performance of your program on the validation set for individual tasks.
  - Supporting diagrams, figures, images that help describe your programs clearly.
  - References that your implementation is based on, or inspired from.

Your report will be marked based on: 1) the clarity and presentation(20%); 2) the proposed methods and the judgements of your design(50%); and 3) experimental results on the validation images and discussions(30%). In your report, you can also report multiple different pipelines you have implemented, even they may not work well, and compare their performances.

4.2 Implementation: 50 marks (Task 1: 20 marks; Task 2: 30 marks)

Your implementation will be marked based on the quality of your code (30%) and the testing performance (70%). Your codes are expected to be well written with comments and good structures. If you have implemented multiple pipelines, please specify the best pipeline. Only the specified pipeline will be used to test the performance which will be used to give marks.

Important: *For each task, the marks will be given based on the outputs within 5 minutes of running your program for that particular task. If your program cannot complete the task after a period of **5 minutes**, the program will be terminated.*

4.2.1 Evaluation Environment

- Your implementation will be tested using a virtual machine, which will be provided for your practice.
- Programming language: Java or Python (only the versions found in the provided virtual machine are accepted).
- Datasets for training: three datasets for you to build your model and test your programs will be uploaded in Blackboard: one for the building signages, one for the directional signages, and one for the recognition of digits and directional signs.
- Testing data sets: six images for each task will be used to test the performances of your programs.
- Validation data sets: a separate validation set for each task that simulate the test images - so that you can estimate how your program will perform - will be given. Check for announcements on Blackboard. You will need to test your programs on the validation data sets, and include the results in your reports.

4.2.2 Your electronic submission

- Your electronic submission to Blackboard must be a compressed file (zip) with the following naming convention

  [surname]_[given names]_[student ID].zip

  For example, if your name is Mike Jordan and your student ID is 123456 then your compressed filename is

  `jordan_mike_123456.zip`

- Your files will be uncompressed to the following directory (suppose that you are Mike Jordan as shown above)

  `/home/student/jordan_mike_123456`

  I will refer to this top directory as `$SUBMISSION_DIR` hereinafter.

- In this directory, I will expect to find two Bash scripts
  - `task1.sh`
  - `task2.sh`

  These scripts must set up suitable environment variables and invoke your programs to perform the corresponding tasks. Do not expect the virtual machine for testing your assignment to have the same `bashrc` file and/or other environment variables as found on your own copy of the VM! Make sure that these scripts can be readily executed by the user `student`.

- Your electronic submission should contain only
  - Source code
  - Bash scripts (described above)
  - Compiled programs (possibly more relevant for people using Java)

  - Necessary files for your models

To save space, your submission must not contain

  - Temporary files that are not necessary for running your programs
  - The original training images.

4.2.3 Input

Important: *Input images are placed under specific directories outlined below. Use absolute path to retrieve them.*

Testing Images: The images used for testing your implementation will be located in the following directory:

`/home/student/test`

Under this directory, the sub-directories for each task are as follows

- Task 1: there are six JPEG image files `test01.jpg`,..., `test06.jpg` under

  `/home/student/test/task1`

- Task 2: there are six JPEG image files `test01.jpg`,..., `test06.jpg` under

  `/home/student/test/task2`

The images used for testing will not be available until all the students have submitted their assignments. To test whether your implementation works as expected, you can copy the images used for validation into the test directories and change the file names accordingly.

The performance of your implementation will be tested only on the images for test. However, the performances of your programs on the images for training and validation may be tested to verify the accuracy of your reports.

In case that your programs need the images for training and validation to test their performances on the testing images, please refer to the following directories for these images

Training Images: The training images provided in Black Board will be stored in the VM under the directory

`/home/student/train`

Under this directory, the sub-directories for each task are as follows

- Task 1: there are ten JPEG image files `BS01.jpg`,..., `BS10.jpg` under

  `/home/student/train/task1`

- Task 2: there are 12 JPEG image files `DS01.jpg`,..., `DS12.jpg` under

  `/home/student/train/task2`

Validation Images: The validation images provided in Black Board will be stored in the VM under the directory

`/home/student/val`

Under this directory, the sub-directories for each task are as follows

- Task 1: there are six JPEG image files `val01.jpg`,..., `val06.jpg` under

  `/home/student/val/task1`

- Task 2: there are six JPEG image files `val01.jpg`,..., `val06.jpg` under

  `/home/student/val/task2`

4.2.4 Output

Important: The output files produced by your programs should be placed under the `output` in **your** submission directory `$SUBMISSION_DIR/output/`, e.g.

`/home/student/jordan_mike_123456/output`

Specifically, the output files for each task should be produced as follows:

- Task 1: All output files should be written to `$SUBMISSION_DIR/output/task1`. For every input image `testX.jpg` your program must produce
  - An image file `DetectedAreaX.jpg` that shows only the detected area containing the building number.
  - A text file `BuildingX.txt` which reports the building number, e.g. ”Building 303”.
- Task 2: All output files should be written to `$SUBMISSION_DIR/output/task2`. For every input image `testX.jpg` your program must produce
  - An image file `DetectedAreaX.jpg` that shows only the detected area containing all the building numbers.
  - A text file `BuildingListX.txt` which reports a list of the building numbers and directions, e.g. ”Building 201 to the left; Building 209 to the left; ...; Building 101 to the right”.

4.2.5 Your demonstration

A demonstration session will be conducted during a practical to verify your submission. You will be asked questions about your programs. The purpose of this demonstration is to make sure that your submission is your own work and you know exactly what you are doing.

5 Submission

You are required to submit your assignment, including your written report and source code, by Friday 11-Oct-2019, 5:00pm Perth time (Week 11).

Upload your submission electronically via Blackboard, under the Assessments section.

You are responsible for ensuring that your submission is correct and not corrupted. You may make multiple submissions, but only your newest submission will be marked.

You will need to make yourself available for the demonstration session. Exact date and time will be announced on Blackboard.

The late submission policy (see the Unit Outline) will be strictly enforced. A submission 1 second late, according to Blackboard, will be considered 1 day late. A submission 24 hours and 1 second late will be considered 2 days late, and so on.

You must also submit a completed, signed **“Declaration of Originality”** form.

6 Academic Misconduct – Plagiarism and Collusion

Please note the following, which is standard across all units in the department:

Copying material (from other students, websites or other sources) and presenting it as your own work is plagiarism. Even with your own (possibly extensive) modifications,it is still plagiarism.

Exchanging assignment solutions, or parts thereof, with other students is collusion. Engaging in such activities may lead to a grade of ANN (Result Annulled Due to Academic Misconduct) being awarded for the unit, or other penalties. Serious or repeated offences may result in termination or expulsion.

You are expected to understand this at all times, across all your university studies, with or without warnings like this.

END OF ASSIGNMENT.