Machine Perception
COMP3007

ASSIGNMENT

Due Date: Week 11 - Friday 4 October 2024 at 5pm.
Weight: 40% of the unit mark.

Note: *This document is subject to minor 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.*

1 Overview

The purpose of this assignment is to learn practical aspects of developing a machine perception application. You are required to synthesise the theoretical concepts covered in lectures with independent research into a specific machine perception problem. Your task is to apply the skills acquired through practical exercises to develop the essential components of a computer vision-based barcode number recognition system. This assignment not only demands technical proficiency in programming but also seeks to enhance your communication skills, as you will be expected to clearly articulate your approach, methodology, and findings.

2 Background

In this assignment, you will develop a machine perception program to detect and recognise the numbers associated with EAN-13 barcodes from images of products. A barcode is a specialised representation of data in the form of a printed pattern on a product, which is machine-readable and enables the rapid identification of that product.

There are various types of barcodes, each serving different purposes. In this assignment, we will focus specifically on the EAN-13 barcode type. For more details, refer to

`https://en.wikipedia.org/wiki/International_Article_Number#How_the_13-digit_EAN-13_is_encoded`

Essentially, the EAN-13 barcode comprises bars of varying widths, arranged in a specific sequence to encode a number that identifies the product. According to the EAN-13 specification, a barcode number consists of 13 digits and is divided into three groups: the first digit, a left group of six digits, and a right group of six digits. Below the EAN-13 bars, the barcode number is printed, allowing manual entry of

the product ID in case the barcode scan fails.

Although barcodes are typically read using an optical device known as a barcode scanner (which electronically reads and decodes the patterns), computer vision-based barcode readers are also widely available and generally operate on the same decoding principle. This assignment explores an alternative approach to the detection and recognition of barcode numbers *without decoding*. This method could potentially serve as a backup option for conventional computer vision-based barcode readers when the standard decoding approach fails.

A typical machine perception approach to this problem would involve at least the following steps:

- Reading the input image;
- Performing necessary image processing operations;
- Detecting and localising the barcode number, if present;
- Segmenting the barcode number into individual digits;
- Recognising individual digits; and
- Optionally, post-processing the detection results.

You are required to implement an appropriate machine perception pipeline to perform the above steps by developing Python programs. Your implementation will be primarily evaluated based on the following tasks:

- Task 1: Barcode number detection and localisation;
- Task 2: Barcode digit extraction;
- Task 3: Barcode digit recognition;
- Task 4: Complete pipeline.

You are expected to apply the theory discussed in lectures and the skills acquired through practical exercises to complete these tasks. Additionally, you will need to conduct independent research to explore various approaches for solving each task and make informed decisions regarding your implementation and/or innovative solutions tailored to the specific requirements of this assignment.

Each individual task, along with the complete pipeline, will be assessed, and marks will be allocated accordingly. Details of the mark distribution are provided in the subsequent sections.

3 The Tasks

3.1 Task 1: Barcode number detection and localisation

Develop a Python program that reads colour images from a designated directory `img1.jpg`, `img2.jpg`, etc. For each image, the program should determine if a barcode number is present and, if so, extract the region containing the barcode number. Be aware that some test images may be negative, meaning they do not contain a valid EAN-13 barcode number. Your program will be deemed functional if it adheres to the following requirements:

- For a *negative* input image, the program should indicate that no barcode number was found and should not produce any output image or text file.

- For a *positive* input image, the program should output the detected region with coordinates $x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4$ of the barcode number in the original input image, to a text file for every input image to the current working directory.

```
x1,y1,x2,y2,x3,y3,x4,y4
```

  The name of the output file must match the input file, e.g. `img1.txt`, `img2.txt`, etc. A detection is considered successful if 1) the detected area contains all the barcode digits; and 2) the Intersection of Union (IoU) is at least 50%. The groundtruth bounding box is the tightest rectangle containing all the barcode digits.

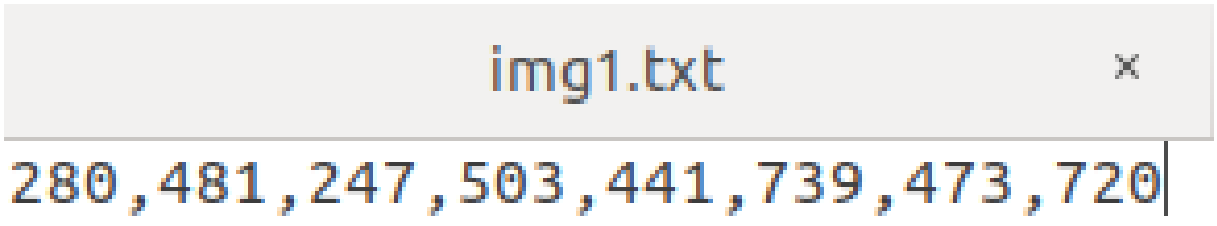
  In addition, the program should also output the detected region, after a suitable geometric transformation, as an image file matching the filename of the input image, e.g. `img1.png`, `img2.png`, etc.

  Please note that each positive test image will contain only one barcode number. Consequently, your program *should not report more than one detected region*. Failure to comply with this will result in a failed detection.

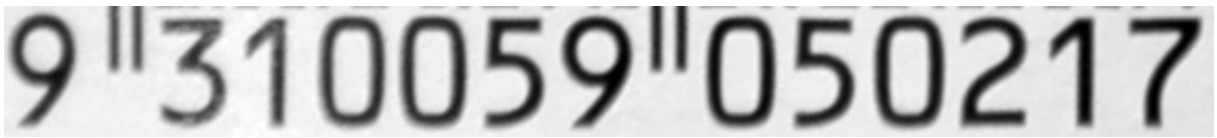
  See Fig. 1 for an example of a positive input image and the required output files.

(a) Input Image `img1.jpg`

(b) Detected Area



(c) Output: Detected coordinates, name the text file as `img1.txt`,`img2.txt`, etc.



(d) Output: Detected area, name the output image file as `img1.png`, `img2.png`, etc.

Figure 1: Task 1

3.2 Task 2: Barcode fdigit image extraction

For this task, you are provided with images of the detected regions containing barcode numbers, as identified in Task 1. You may assume that the detected region has been geometrically transformed if required. Write a Python program to extract individual digits as text files `d01.txt`, `d02.txt`, etc. that contains their top left and bottom right coordinates $x1, y_1, x_2, y_2$. (see Fig. 2).

Your program must also output the segmented digits as image files `d01.png`, `d02.png`, etc. A digit is considered to be successfully segmented if 1) the segmented area contains the correct digit; and 2) the Intersection of Union (IoU) is at least 50%. The groundtruth bounding box is the tightest rectangle containing the digit.

Keep in mind that each EAN-13 barcode number consists of exactly 13 digits, so your program is expected to produce 13 output images for every input image. If your program over-segments the digits, only the first 13 images will be considered for marking.

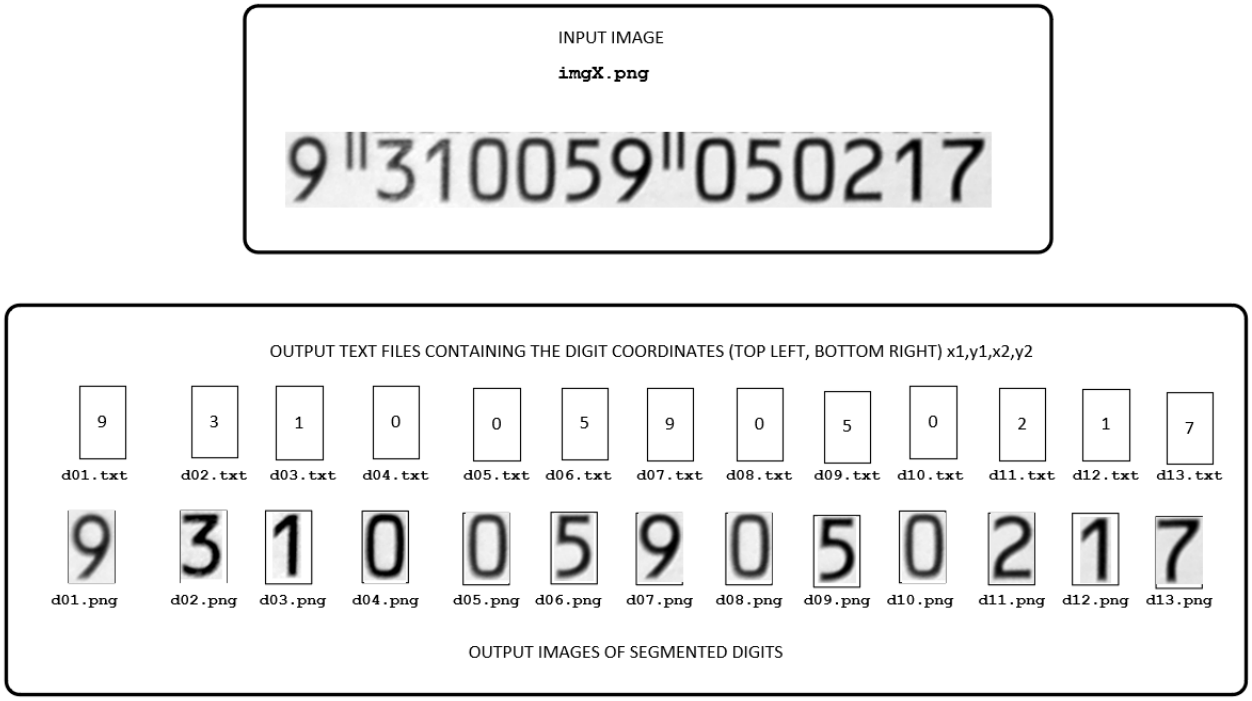
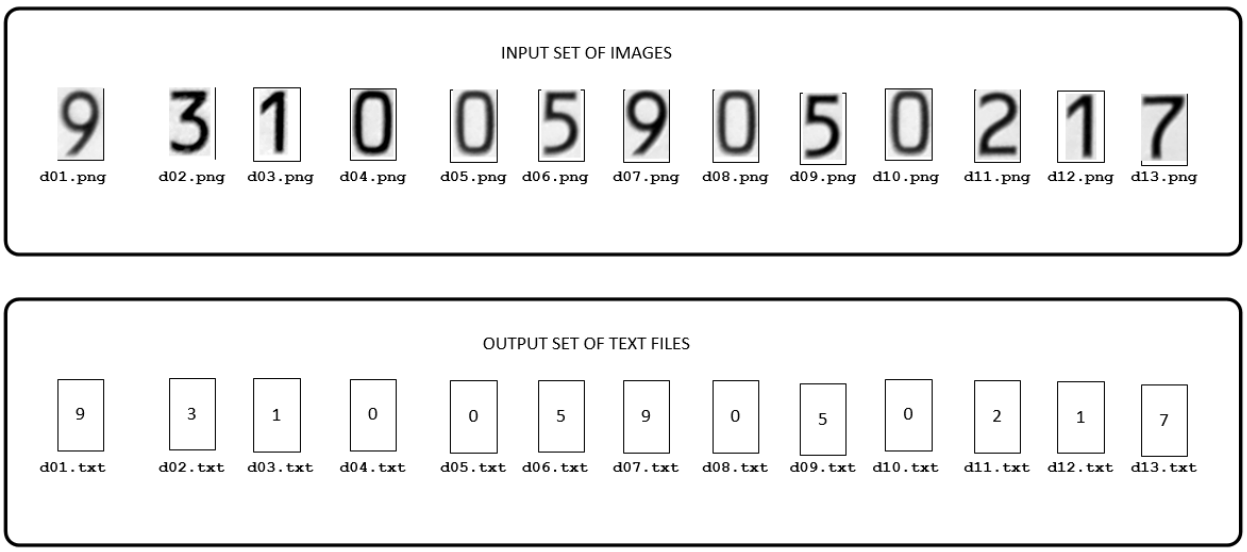


Figure 2: Task 2

3.3 Task 3: Barcode number recognition

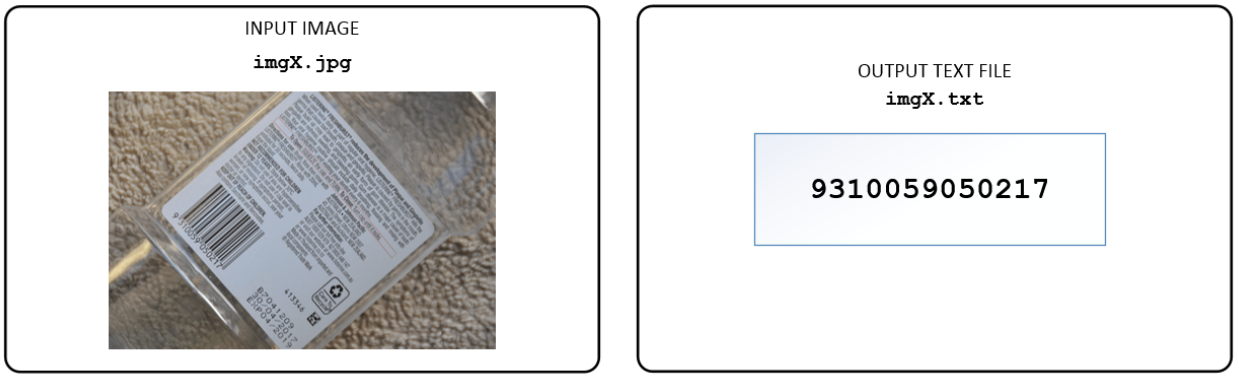
In this task, you are given sets of images of individual digits that are found from Task 2. Each set of images corresponds to a barcode number. Write a program that recognises the digits from each set and, for **each** input digit image, outputs the recognized digit as a text file (see the following figure).



3.4 Task 4: Complete perception pipeline

In this task, you complete the perception pipeline by combining the individual programs that you have implemented for the previous tasks. For a given input image

- If it is a negative image, your program must produce nothing;
- If it is a positive image, your program must produce a single text file containing the barcode number found.



4 Specifications

4.1 Environment

- Programming language: Python. Your programs must run in Python 3.10.12.
- Allowed libraries: Please refer to Blackboard for the most up-to-date information.
- Validation set: A separate validation set, which simulates the test images and allows you to estimate how your program will perform, is provided on Blackboard. You will need to test your programs on this validation set and include the results in your report.
- Students must implement their Python programs using an assignment Python framework developed specifically for this assignment. This framework contains skeleton code and specific

structures to facilitate effective marking of your implementation. Please see details on Blackboard.

4.2 Images and Barcode Numbers

- *All EAN-13 barcode numbers in this assignment start with digit 9.*
- Every positive image containing an EAN-13 barcode can be taken in either
  - Case 1: the barcode plane is (approximately) orthogonal to the camera lens axis, and there are possible 8 orientations as shown in the figure to follow

  - Case 2: the camera lens axis is at an angle to the barcode plane as shown in the figure below

- The negative images may contain incomplete barcodes or barcodes without the barcode numbers (see below)

- The actual test set used for marking your assignment will contain
  - Four (4) positive images
  - One (1) negative image

  Image sets derived from these 5 images for assessing each individual task will be produced accordingly.

4.3 Pass Requirements

Your implementation must do better than random guessing before marks could be given. Specifically,

- Task 1: your program must detect and/or localize correctly at least one positive image;

- Task 2: your program must extract correctly at least 25% of the total number of digits across all barcode numbers.
- Task 3: your program must recognize correctly at least 25% of the total number of digits across all barcode numbers.
- Task 4: your program must produce ‘‘satisfactory” recognition of at least one positive image. Here, ‘‘satisfactory” recognition means at least 10 out of 13 digits are correctly recognized and ordered.

4.3.1 Reasonable Performance

Your electronic submission must contain the source code, trained models, and run programs written for the tasks. Each of your programs must output messages indicating what it is currently doing. For each task, if your program does not produce any results after a period of **5 minutes** (regardless of whether the eventual results are correct or not), it will be considered failed and you will get no marks for that particular task.

4.4 Programs, Directories, and Files

Please follow the following convention for files and directories of your submission:

4.4.1 Your electronic submission of the Python implementation

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

  [surname]_[student ID].zip

  For example, if your name is Donald Trump and your student ID is 12345678 then your compressed filename is

  `trump_12345678.zip`

- Your electronic submission should contain only
  - Source code
  - Necessary files for your models

  To save space, your submission must not contain temporary files that are not necessary for running your programs.
- Your zip file must not exceed 50 MB!

4.4.2 Input

Important: All test images are placed in a dedicated directory on a Linux machine, which I will refer to as $TEST_DIR. The marking script will automatically pass the path to this directory to your Python program. Under this directory, the images are structured as follows

- Task 1: there are five JPEG image files `img1.jpg`,..., `img5.jpg` under

  `$TEST_DIR/task1`

- Task 2: there are four PNG image files `barcode1.png`, ..., `barcode4.png` under

  `$TEST_DIR/task2`

- Task 3: there are four sub-directories, each corresponding to the number plate in Task 2

  `$TEST_DIR/task3/barcode1`
  `$TEST_DIR/task3/barcode2`
  `$TEST_DIR/task3/barcode3`
  `$TEST_DIR/task3/barcode4`

  In each sub-directory, there are 13 PNG image files `d01.png`, `d02.png`, ..., `d13.png` ..., each corresponds to a segmented digit of the related EAN-13 barcode number.

- Task 4: the input images are the same as those found in Task 1.

4.4.3 Output

Important: The output files produced by your programs should be placed under the `output` in a directory which I refer to as `$SUBMISSION_DIR/output/`, where `$SUBMISSION_DIR` is where your submission is unpacked to.

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

- Task 1: All output files should be written to `$SUBMISSION_DIR/output/task1`. For every *positive* input image `imgX.jpg` your program must produce an image file `barcodeX.png` that shows only the detected area containing the barcode number. For the negative input image, your program should produce nothing. In addition, your program must also output the coordinates of the detected area in the original image, e.g. `imgX.txt`.

- Task 2: the image files for individual 13 segmented digits must be named `d01.png`,`d02.png`, ...,`d13.png` following the exact order how they appear in the barcode number, and for input image `barcodeX.png`, they must be located under

  `$SUBMISSION_DIR/output/task2/barcodeX/`

- Task 3: your program must output a text file `dYY.txt` for every individual input digit image file `dYY.png` and for every barcode number under

  `$SUBMISSION_DIR/output/task3/barcodeX/`

  Each text file must contain only the recognized digit.

- Task 4: it must produce a text file `imgX.txt` for every *positive* input image `imgX.jpg` and place it under

  `$SUBMISSION_DIR/output/task4/`

  Each text file must contain **all** digits in the exact order how they appear in the barcode number. Your program must produce nothing if it is a negative input image.

4.5 Your report

A written report must also be submitted to Blackboard via a separate submission link. The report must contain

- A completed assignment cover sheet
- The following information
  - 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.
- The report must also follow a similar naming convention and is in PDF. For example, if your name is Donald Trump and your student ID is 12345678 then your report should be named

`trump_12345678.pdf`

4.6 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 Marking Detail

The total mark of this assignment is 100, and it is distributed as follows

5.1 Satisfactory Submission: 25 Marks

A maximum 25 marks may be awarded for the following

- Your submission meets the deadline and follows the instruction.
- Your source code is well structured and fully commented, indicating good programming practice.
- All developed programs meet the ‘‘Reasonable Performance” criteria.
- A complete report outlining your design, your implementation, and performance of your program on the validation set.
- Your submission is your own work and you understand what you are doing.

Your actual marks will be determined somewhat *subjectively* by the marker based on inspection of your programs, your code, your report, and the interview.

5.2 Task 1: 20 Marks

- Zero marks if your program does not meet the the minimum pass requirements for Task 1
- 12 marks if your program meets the the minimum pass requirements for Task 1, and additional marks will be given (up to 8 marks) for extra barcode numbers correctly detected.

5.3 Task 2: 20 Marks

- Zero marks if your program does not meet the the minimum pass requirements for Task 2
- 12 marks if your program meets the the minimum pass requirements for Task 2, and additional marks will be given (up to 8 marks) for extra barcode digits correctly extracted.

5.4 Task 3: 20 Marks

- Zero marks if your program does not meet the the minimum pass requirements for Task 3
- 12 marks if your program meets the the minimum pass requirements for Task 3, and additional marks will be given (up to 8 marks) for extra barcode digits correctly recognized.

5.5 Task 4: 15 Marks

- Zero marks if your program does not meet the the minimum pass requirements for Task 4
- 5 marks if your program meets the the minimum pass requirements for Task 4, and additional marks will be given (up to 10 marks) for extra barcode number satisfactorily recognized, this means
  - The negative image is detect correctly
  - At least 10 out of 13 digits of a positive image are correctly recognized

NOTE Your total score for this assignment will need to be at least **30 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 tests and final exam.

6 Submission

You are required to submit your assignment by Week 11 - Friday 4 October 2024 at 5pm.

Your Python implementation must be submitted electronically via Blackboard as a zip file.

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.

A written report must be submitted electronically via a separate submission link. See above for detail.

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

The late submission policy (see the Unit Outline) will be strictly enforced.

7 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.

Generative AI

Whilst using Generative AI to help you learn is acceptable, what you submit must be your own work. You must understand what the code you submit does. You will be asked to sit an interview and if you do not demonstrate an understanding of your submission, you may receive a zero mark for the submission and this may lead to you failing the assignment, and hence the unit. The bottom line is that you have to fully understand the code that you submit.

END OF ASSIGNMENT