UTM
UNIVERSITI TEKNOLOGI MALAYSIA
Malaysia-Japan
International
Institute of Technology
(MJIIT)

MALAYSIA-JAPAN INTERNATIONAL INSTITUTE OF TECHNOLOGY
DEPARTMENT OF ELECTRONIC SYSTEM ENGINEERING

SMJE 4383 ADVANCED PROGRAMMING
SESSION 2022/2023-1

ASSIGNMENT 1

	NAME	MATRICS NO
GROUP MEMEBRS	MOHAMAD FARIZUAN AKMAL BIN JAMALUDIN	A19MJ0049
	MUHAMMAD DANIAL FIKRI BIN KAMARUZAMAN	A19MJ0066
LECTURE	DR ZOOL HILMI BIN ISMAIL	

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 Background Study

Screen scraping is a technique of copying information from a screen display. Screen scraping read text data from display screen. Its extract information from the visual display onto raw text. The extracted data can then be stored and analysed, or used to perform other tasks. Screen scraping is commonly used to gather information from websites. Screen scraping can be done by writing custom code using programming languages like Python or using specialized screen scraping tools or libraries.

OCR stands for Optical Character Recognition. OCR is a process of converting an image into a machine-readable text format by a computer. OCR will recognize a text in the image and process text that is contained within an image. The extracted text can be useful to edit, analyse and stored the text. In this digital era, OCR are widely use such as document scanning and digitization. Another application of OCR is in the smartphone, where the photos will detect the readable text in the images so that users can copy the text for another purpose. Some of factors that might impact the accuracy of OCR is the quality of the image, the size and style of the text, and the complexity of the background.

By using OCR, we can do the screen scraping technique to extract the machine-readable text from the User Interface (UI) of the application or images. At first, The screen scraper would capture an image of the necessary area of the web page before passing it to the OCR engine for processing. Then, the OCR engine will analyse the image and do recognize and extract the text in the image. The extracted text then will be returned to the screen scraper for storage, analysis or further processing.

1.2 Project Framework

This project will be conducted by using the Python3 on Ubuntu operating system. In order to use OCR and screen scraping, python language can be use for that purpose as it allows us to install a library that can be use for the screen scraping and OCR.

For screen scraping, we will be using the cv2 module in the OpenCV. Cv2 provides functions for loading and saving images, drawing shapes and text on images, and performing operations such as edge detection and thresholding. In our project, cv2 will recognize the car number plate part including the number plate shape that contain character and numbers. The extracted part of the images then will be used for OCR.

For the OCR, the tesseract library. Tesseract is OCR engine developed by Google. Tesseract is an open-source software library that can be use for extracting text from pictures. One of the advantage of tesseract OCR engine is it support a wide range of languages including English, France, German and many more. In our project, after the cv2 recognize part of the number plate, tesseract extract the number plate and convert it to a machine-readable text format.

In summary, by using the combination of cv2 and tesseract we will be able to recognize the car number plate and print it on the text format. The cv2 will show us the original image and also the image of number plate part extracted by the cv2. On Ubuntu terminal, the extracted text from the images will be print on the terminal.

1.3 Problem Statement

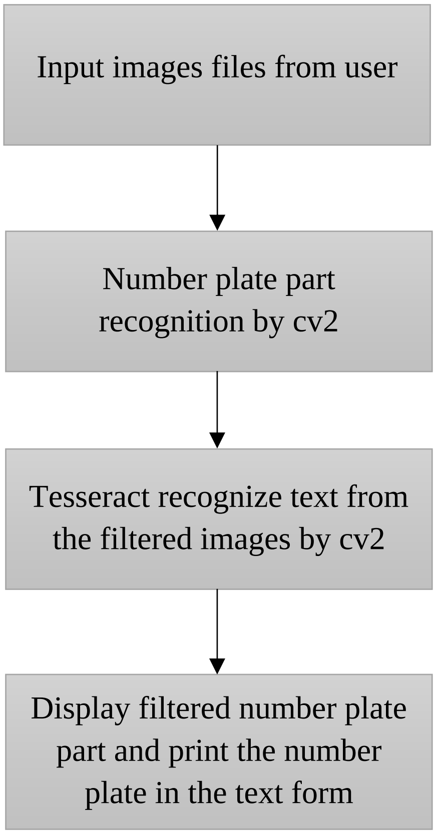
Based on the instruction given, student need to carry out a programming task that can execute end to end process for Screen Scraping OCR Text Recognition using Python Script This topic is selected because it is part of the academia industrial collaboration at MJIIT.

1.4 Objective

1. To implement Python script that can process for Screen Scraping and OCR text recognition
2. To extract the text information from the images using OCR and display it to the user.

CHAPTER 2: METHODOLOGY

2.1 Project Workflow

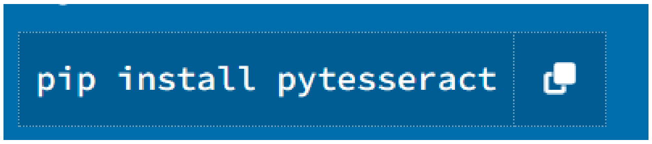

Input images files from user
Number plate part recognition by cv2
Tesseract recognize text from the filtered images by cv2
Display filtered number plate part and print the number plate in the text form

Above is the project workflow that conclude how the cv2 and tesseract library is used and how the output of the process is display to the user. Cv2 is mainly used to detect the number plate then from that tesseract use the information extracted by cv2 to convert it to machine readable text format.

2.2 Library used

1. Tesseract:

   Python-tesseract is a Python-based optical character recognition (OCR) tool. In other words, it will recognise and "read" text encoded in photos. In order to install this library in Python environment, this command can be use:

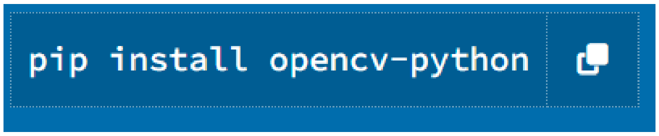


```
pip install pytesseract
```

Figure 1: Tesseract installation command

2. Cv2 (OpenCV):

   This cv2 is use for image processing purposes and it is easy to use by using Python. In order to setup OpenCV in Python environment, this command can be use:



```
pip install opencv-python
```

Figure 2: OpenCV installation command

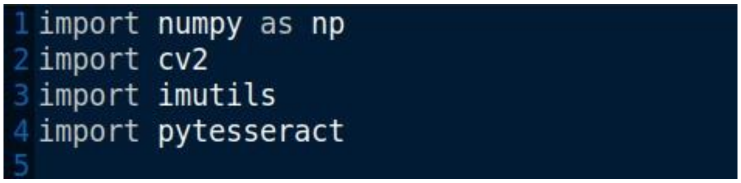
3. Numpy:

   NumPy is a library for the Python programming language that provides support for arrays and matrices. NumPy can be used for mathematical calculation.

4. Imutils:

   Functions for OpenCV that make simple image processing tasks like translation, rotation, scaling, skeletonization, showing Matplotlib pictures, sorting contours, detecting edges, and much easier.

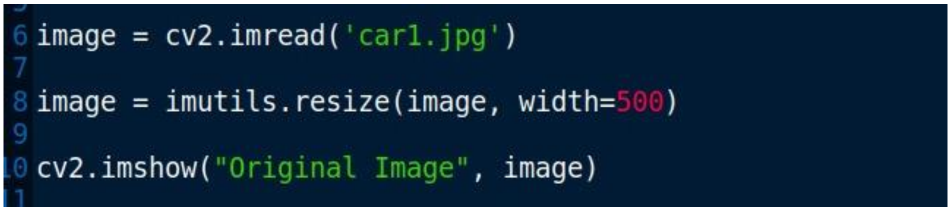
2.3 Python Codes



```
1 import numpy as np
2 import cv2
3 import imutils
4 import pytesseract
5
```

Figure 3: coding line 1 to line 4

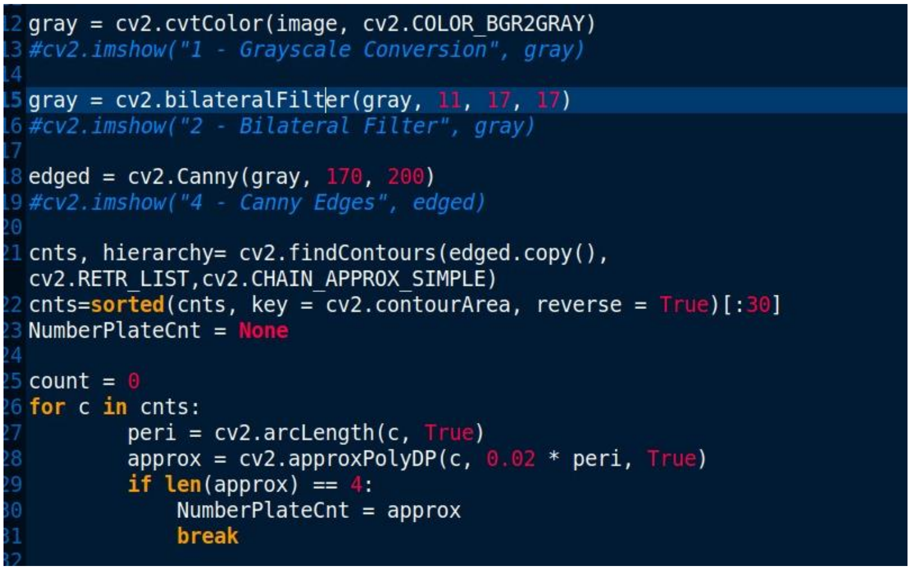
On the top of code, we include all of the libraries that will be used for this program which numpy, cv2, imutils and tesseract as shown in Figure 3.



```
6 image = cv2.imread('car1.jpg')
7
8 image = imutils.resize(image, width=500)
9
10 cv2.imshow("Original Image", image)
11
```

Figure 4: coding line 6 to line 10

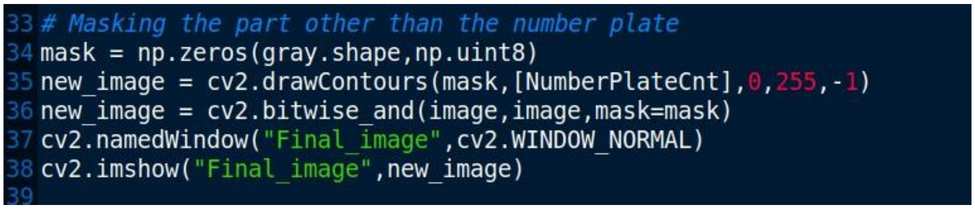
In Figure 4, the code shown will read and open the image file based on the file name that was set by the user. In our program, we want the code the extract text from file ‘car1.jpg’ file. Then, imutils library is used to resize the image. Cv2 will pop up a window with a title of “Original Image” which will shows the image that we want the code to read.



```
12 gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
13 #cv2.imshow("1 - Grayscale Conversion", gray)
14
15 gray = cv2.bilateralFilter(gray, 11, 17, 17)
16 #cv2.imshow("2 - Bilateral Filter", gray)
17
18 edged = cv2.Canny(gray, 170, 200)
19 #cv2.imshow("4 - Canny Edges", edged)
20
21 cnts, hierarchy= cv2.findContours(edged.copy(),
   cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)
22 cnts=sorted(cnts, key = cv2.contourArea, reverse = True)[:30]
23 NumberPlateCnt = None
24
25 count = 0
26 for c in cnts:
27         peri = cv2.arcLength(c, True)
28         approx = cv2.approxPolyDP(c, 0.02 * peri, True)
29         if len(approx) == 4:
30             NumberPlateCnt = approx
31             break
32
```

Figure 5: Coding line 12 to line 31

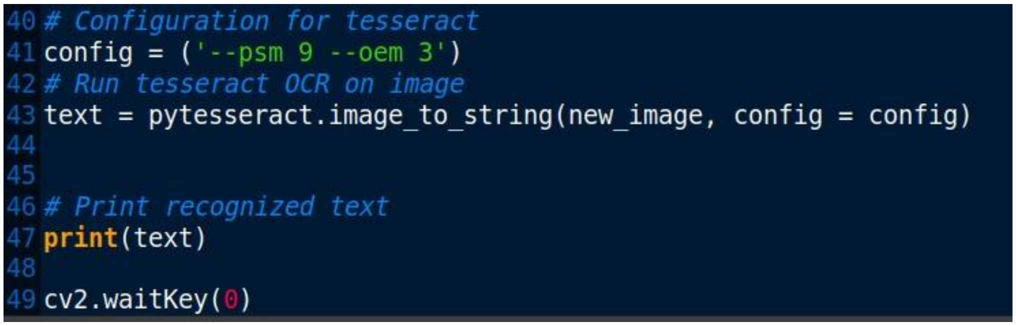
Figure 5 shows the code where the image filtering is being process. The image will detect only the part of the number plate and use only that part for OCR.



```
33 # Masking the part other than the number plate
34 mask = np.zeros(gray.shape,np.uint8)
35 new_image = cv2.drawContours(mask,[NumberPlateCnt],0,255,-1)
36 new_image = cv2.bitwise_and(image,image,mask=mask)
37 cv2.namedWindow("Final_image",cv2.WINDOW_NORMAL)
38 cv2.imshow("Final_image",new_image)
39
```

Figure 6: Coding line 33 to line 38

On the Figure 6, we can the code where the program will be masking the part other than the number plate. This means that the output images will be only the number plate part. Cv2 then will pop up a new window with a title of “Final_image” that will show the number plate image after part other than the number plate is masked.

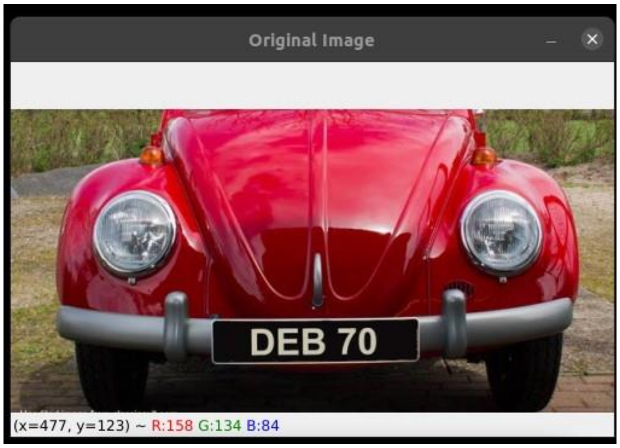


```
40 # Configuration for tesseract
41 config = ('--psm 9 --oem 3')
42 # Run tesseract OCR on image
43 text = pytesseract.image_to_string(new_image, config = config)
44
45
46 # Print recognized text
47 print(text)
48
49 cv2.waitKey(0)
```

Figure 7: Coding line 40 to line 49

On Figure 7, OCR process is being done by tesseract. By using the image extracted by cv2 it recognizes the text in the image. For the configuration, we use Page Sementation Modes (psm) of 9 which means that it will treat the image as a single word in a circle and OCR Engine Modes(oem) of 3 which means default, based on what engine is available. After conversion and extraction is done, the recognized text will be print on the terminal.

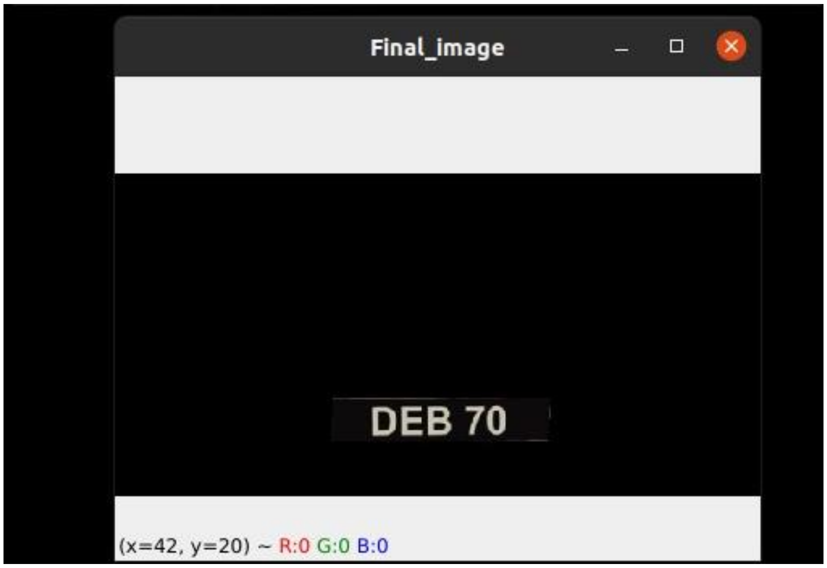
CHAPTER 3: RESULT AND DISCUSSION



(x=477, y=123) ~ R:158 G:134 B:84

Figure 8: Code output on cv2 window

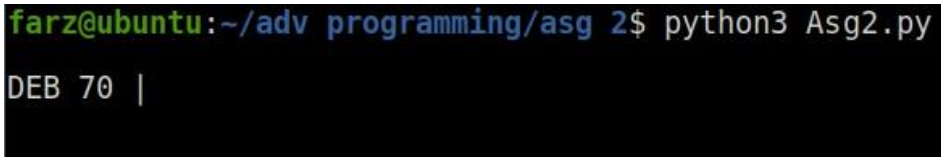
On Figure 8, we can see the pop up window after the code was run in Python environment in Ubuntu Operating System. The window was created by cv2 library, where is shows the original input image and the image after the other part besides the number plate part was masking out.



(x=42, y=20) ~ R:0 G:0 B:0

Figure 9: Final Image window

“Final_image” window is the number plate part that will be use by Tesseract OCR to recognize text in that image as shown in Figure 9.



```
farz@ubuntu:~/adv programming/asg 2$ python3 Asg2.py
DEB 70 |
```

Figure 10: Ubuntu Terminal

On Figure 10, we can the output is print on the Ubuntu terminal. The output is coming from the code where the Tesseract OCR recognize text on final image provide by cv2 library and print the recognized text on the terminal. As discuss earlier, the OCR technology is not 100% accurate, we accuracy can vary depend on the quality of the image, the size and style of the text, and the complexity of the background. As we can see on the result, it detects the number plate white frame as text and print it as “|” character since the cv2 did not get rid of that white frame completely.

CHAPTER 4: CONCLUSION

In conclusion, based on the result that we have obtained, we have finally fulfilled the objective that we want to achieve and conducted the project based on the problem statement stated. Python is a powerful language that can be use for screen scraping and OCR text recognition. The combination of this technology can help to solve the industrial-based problem

References

- Education, I. C. (2022, January 5). *What Is Optical Character Recognition (OCR)?* Retrieved from Ibm.com: https://www.ibm.com/cloud/blog/optical-character-recognition
- Gillis, A. S. (2020). *screen scraping*. Retrieved from Data Center: https://www.techtarget.com/searchdatacenter/definition/screen-scraping
- *Screen Scraping: Middleware's Dirty Little Secret | ARCA*. (2022, October 10). Retrieved from ARCA: https://www.arca.com/resources/screen-scraping-middlewares-dirty-little-secret/#:~:text=Many%20people%20think%20Optical%20Character,from%20an%20active%20application%20window.
- Techopedia. (2012, January 30). *Screen Scraping*. Retrieved from Techopedia.com: https://www.techopedia.com/definition/16597/screen-scraping