Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)

Subject: Image Processing and Computer Vision - II Laboratory (DJ19DSL702)

AY: 2024-25

Bhuvi Ghosh
60009210191

Experiment 4

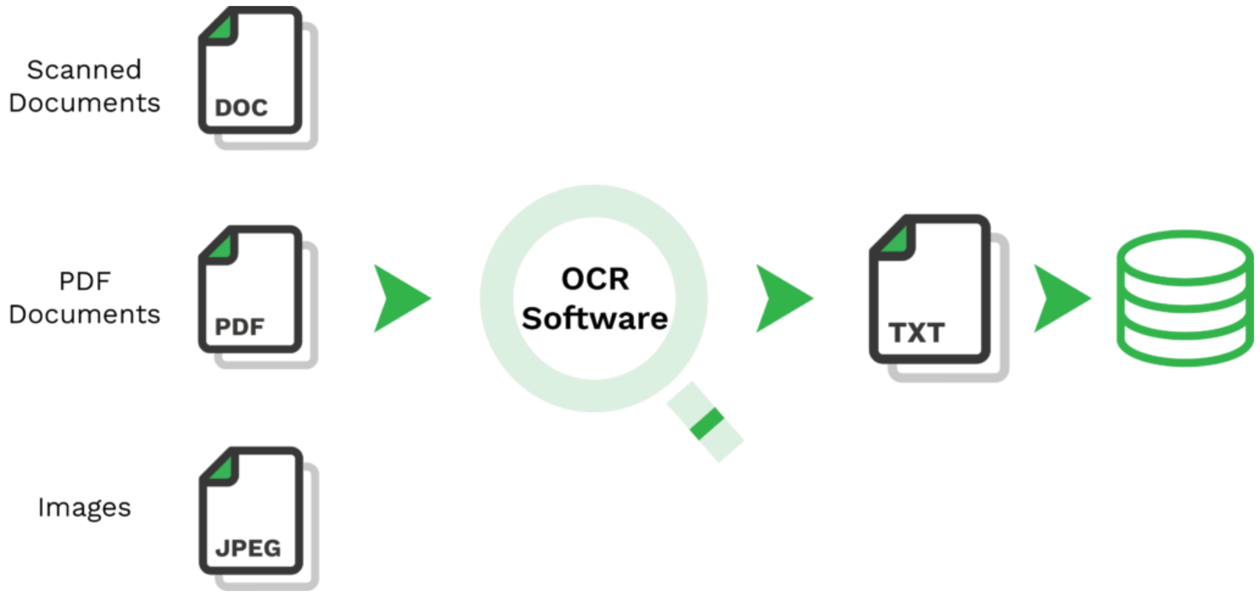
(OCR for Text Recognition)

Aim: Identify text from images using OCR

Theory:

1. Introduction to OCR:

Optical Character Recognition (OCR) is a technology that enables the extraction of textual information from images, scanned documents, or other media containing text. It plays a pivotal role in various applications, including digitization of printed documents, text extraction from images, and enabling text-based search in images.



2. Workflow of OCR:

OCR involves several key steps in its workflow:

a. Preprocessing: This step includes tasks like image resizing, noise reduction, contrast enhancement, and binarization to prepare the image for text extraction.

b. Text Localization: In this step, techniques like text detection and localization are used to identify regions of the image that contain text.

c. Text Segmentation: Text segmentation involves breaking down the identified text regions into individual characters or lines. This step is particularly important for multi-line or multi-column texts.

d. Character Recognition: The recognized text segments are then processed through character recognition algorithms, where each character is identified and converted into machine-readable text.

e. Post-processing: After character recognition, post-processing techniques like spell-checking and error correction are applied to improve the accuracy of the recognized text.

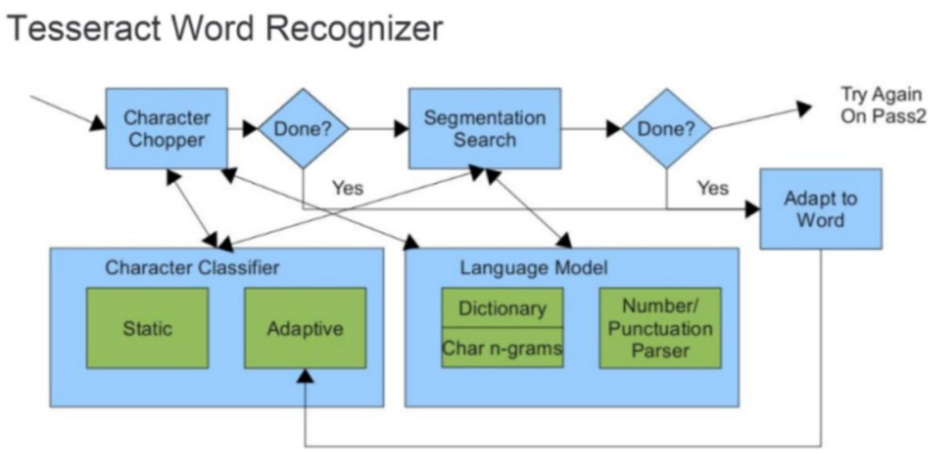
3. Tesseract OCR:

Tesseract is an open-source OCR engine developed by Google. It supports multiple languages and is widely used due to its accuracy and robustness. Tesseract works on the principle of machine learning, utilizing a combination of neural networks and traditional algorithms to recognize text from images.

a. Training Data: Tesseract has been trained on a large dataset of text and corresponding images to learn the patterns of various characters and fonts.

b. Language Models: Tesseract supports various language models that allow it to recognize text in different languages. These models are trained specifically for each language.

c. Usage: Tesseract can be used as a command-line tool or integrated into software applications through its API. It provides options for configuring parameters and improving recognition accuracy.



4. Experiment Setup:

In the lab experiment, students will follow these steps:

a. Image Input: Provide an image containing text as input to the Tesseract OCR engine.

b. Preprocessing: Preprocess the image to enhance text visibility if needed.

c. Text Extraction: Use Tesseract to extract text from the image.

d. Post-processing: Apply post-processing techniques to correct any recognized text errors.

e. Output: Obtain the final recognized text output from the Tesseract OCR engine.

5. Evaluation and Discussion:

After completing the experiment, students should evaluate the accuracy of the extracted text. They can discuss factors affecting accuracy, potential sources of errors, and strategies to improve OCR performance. This can include discussing image quality, font types, languages, and the impact of preprocessing steps on OCR results.

Now, let's move on to the practical part of the lab and apply our knowledge to identify text from images!

Lab Assignments to complete in this session:

Dataset: Use sample images containing text.

The steps involved in the experiment are as follows:

1. Import Tesseract OCR and relevant libraries.

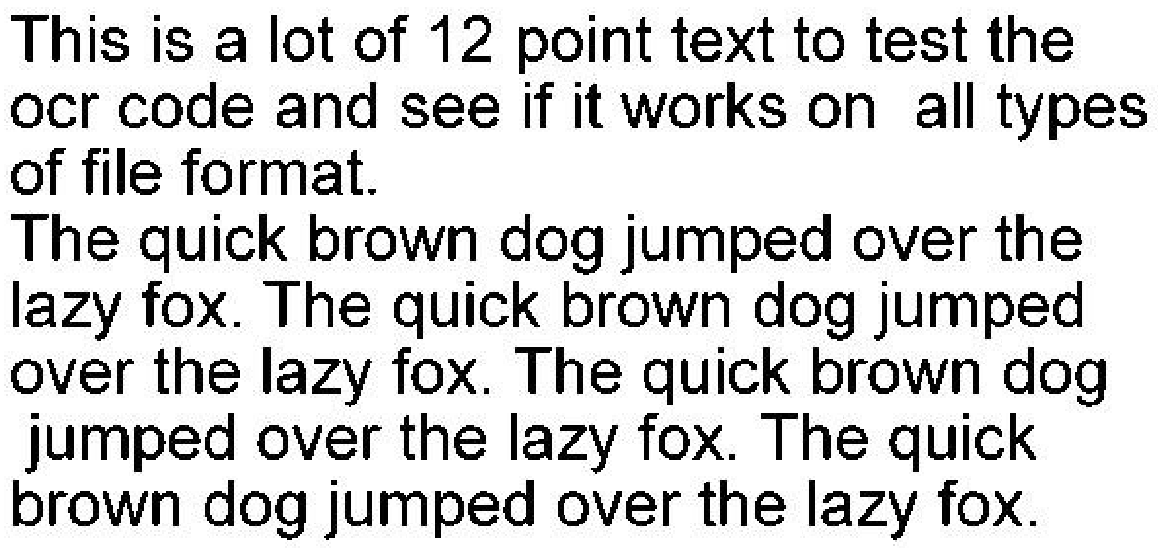
2. Apply Tesseract on the sample images.

```
!sudo apt update
!sudo apt install tesseract-ocr
!sudo apt install libtesseract-dev
```

```
Get:1 https://cloud.r-project.org/bin/linux/ubuntu jammy-cran40/
InRelease [3,622 B]
Hit:2
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/
x86_64  InRelease
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [129
kB]
Ign:4 https://r2u.stat.illinois.edu/ubuntu jammy InRelease
Get:5 https://r2u.stat.illinois.edu/ubuntu jammy Release [5,713 B]
Hit:6 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:7 https://r2u.stat.illinois.edu/ubuntu jammy Release.gpg [793 B]
Get:8 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [128
kB]
Hit:9 https://ppa.launchpadcontent.net/deadsnakes/ppa/ubuntu jammy
InRelease
Hit:10 https://ppa.launchpadcontent.net/graphics-drivers/ppa/ubuntu
jammy InRelease
Hit:11 https://ppa.launchpadcontent.net/ubuntugis/ppa/ubuntu jammy
InRelease
Get:12 http://security.ubuntu.com/ubuntu jammy-security/main amd64
Packages [2,318 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [127
kB]
Get:14 https://r2u.stat.illinois.edu/ubuntu jammy/main all Packages
[8,359 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/universe amd64
Packages [1,156 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64
Packages [2,595 kB]
Get:17 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64
Packages [1,445 kB]
Get:18 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64
Packages [33.7 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64
Packages [81.4 kB]
Get:20 https://r2u.stat.illinois.edu/ubuntu jammy/main amd64 Packages
[2,588 kB]
Fetched 19.0 MB in 12s (1,613 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
54 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: Skipping acquire of configured file 'main/source/Sources' as
repository 'https://r2u.stat.illinois.edu/ubuntu jammy InRelease' does
not seem to provide it (sources.list entry misspelt?)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  tesseract-ocr-eng tesseract-ocr-osd
```

```
The following NEW packages will be installed:
  tesseract-ocr tesseract-ocr-eng tesseract-ocr-osd
0 upgraded, 3 newly installed, 0 to remove and 54 not upgraded.
Need to get 4,816 kB of archives.
After this operation, 15.6 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-
ocr-eng all 1:4.00~git30-7274cfa-1.1 [1,591 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-
ocr-osd all 1:4.00~git30-7274cfa-1.1 [2,990 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-
ocr amd64 4.1.1-2.1build1 [236 kB]
Fetched 4,816 kB in 2s (2,822 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog
based frontend cannot be used. at
/usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78, <> line 3.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package tesseract-ocr-eng.
(Reading database ... 123614 files and directories currently
installed.)
Preparing to unpack .../tesseract-ocr-eng_1%3a4.00~git30-7274cfa-
1.1_all.deb ...
Unpacking tesseract-ocr-eng (1:4.00~git30-7274cfa-1.1) ...
Selecting previously unselected package tesseract-ocr-osd.
Preparing to unpack .../tesseract-ocr-osd_1%3a4.00~git30-7274cfa-
1.1_all.deb ...
Unpacking tesseract-ocr-osd (1:4.00~git30-7274cfa-1.1) ...
Selecting previously unselected package tesseract-ocr.
Preparing to unpack .../tesseract-ocr_4.1.1-2.1build1_amd64.deb ...
Unpacking tesseract-ocr (4.1.1-2.1build1) ...
Setting up tesseract-ocr-eng (1:4.00~git30-7274cfa-1.1) ...
Setting up tesseract-ocr-osd (1:4.00~git30-7274cfa-1.1) ...
Setting up tesseract-ocr (4.1.1-2.1build1) ...
Processing triggers for man-db (2.10.2-1) ...
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libarchive-dev libleptonica-dev
The following NEW packages will be installed:
  libarchive-dev libleptonica-dev libtesseract-dev
0 upgraded, 3 newly installed, 0 to remove and 54 not upgraded.
Need to get 3,744 kB of archives.
After this operation, 16.0 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64
libarchive-dev amd64 3.6.0-1ubuntu1.1 [582 kB]
```

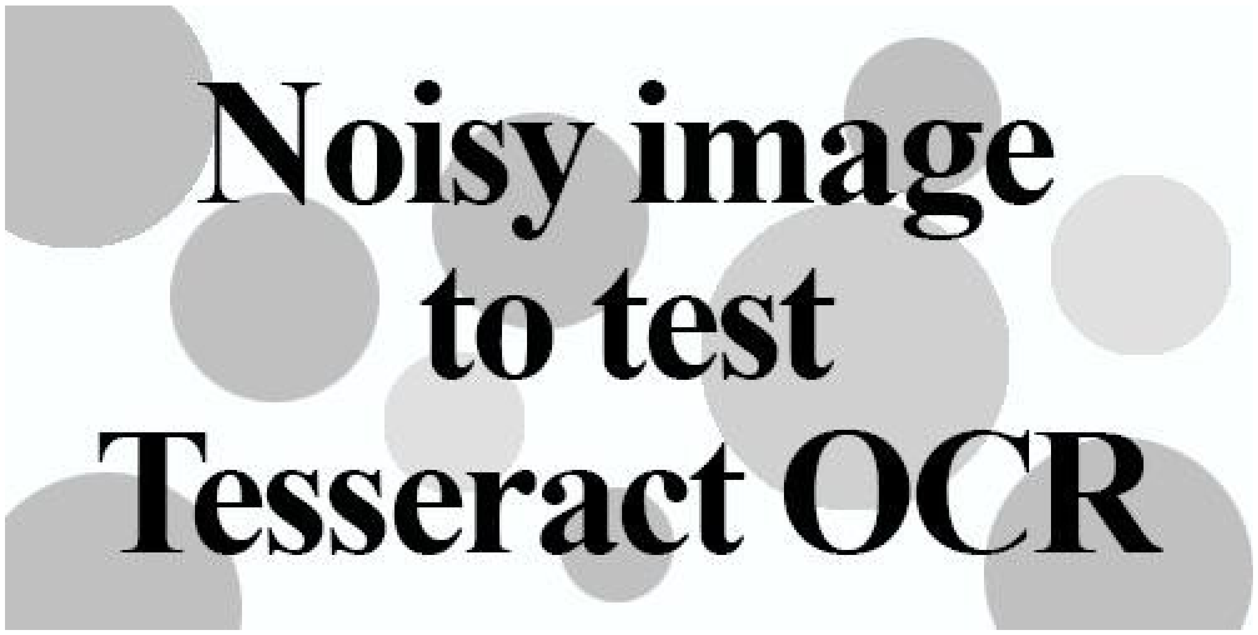
```
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64
libleptonica-dev amd64 1.82.0-3build1 [1,562 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64
libtesseract-dev amd64 4.1.1-2.1build1 [1,600 kB]
Fetched 3,744 kB in 2s (2,265 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog
based frontend cannot be used. at
/usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78, <> line 3.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package libarchive-dev:amd64.
(Reading database ... 123661 files and directories currently
installed.)
Preparing to unpack .../libarchive-dev_3.6.0-1ubuntu1.1_amd64.deb ...
Unpacking libarchive-dev:amd64 (3.6.0-1ubuntu1.1) ...
Selecting previously unselected package libleptonica-dev.
Preparing to unpack .../libleptonica-dev_1.82.0-3build1_amd64.deb ...
Unpacking libleptonica-dev (1.82.0-3build1) ...
Selecting previously unselected package libtesseract-dev:amd64.
Preparing to unpack .../libtesseract-dev_4.1.1-2.1build1_amd64.deb ...
Unpacking libtesseract-dev:amd64 (4.1.1-2.1build1) ...
Setting up libleptonica-dev (1.82.0-3build1) ...
Setting up libarchive-dev:amd64 (3.6.0-1ubuntu1.1) ...
Setting up libtesseract-dev:amd64 (4.1.1-2.1build1) ...
Processing triggers for man-db (2.10.2-1) ...

!pip install pytesseract -qU

import cv2
from google.colab.patches import cv2_imshow
from skimage import io
import pytesseract

img1 =
cv2.cvtColor(io.imread('https://jeroen.github.io/images/testocr.png'),
cv2.COLOR_BGR2RGB)
cv2_imshow(img1)

# Adding custom options
custom_config = r'--oem 3 --psm 6'
print(pytesseract.image_to_string(img1, config=custom_config))
```



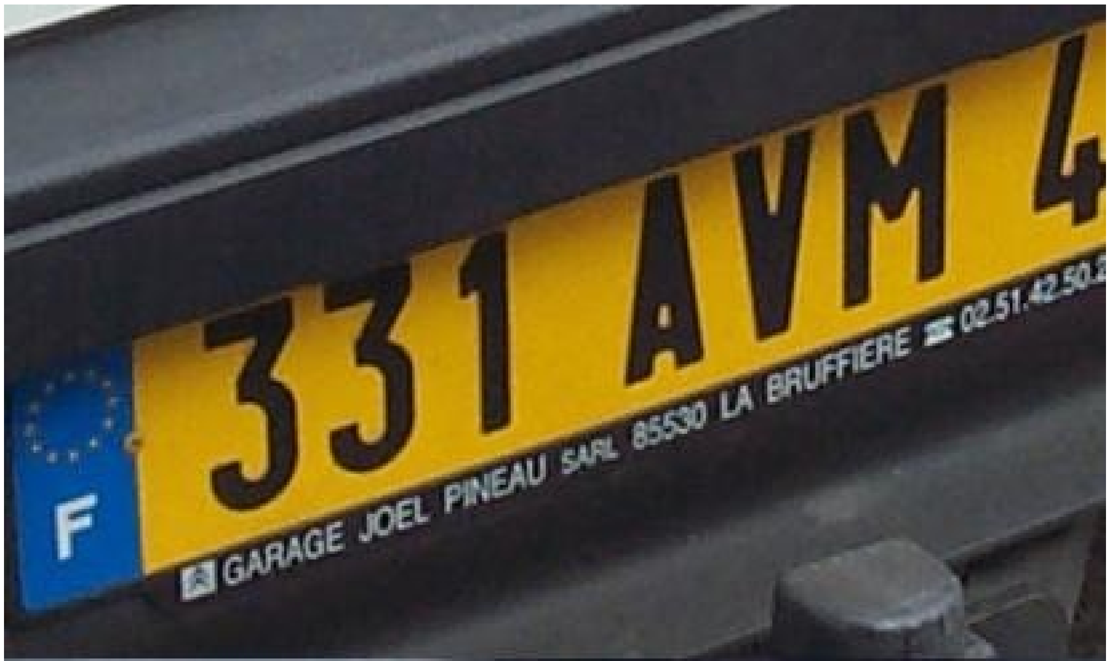
This is a lot of 12 point text to test the ocr code and see if it works on all types of file format.
The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox.

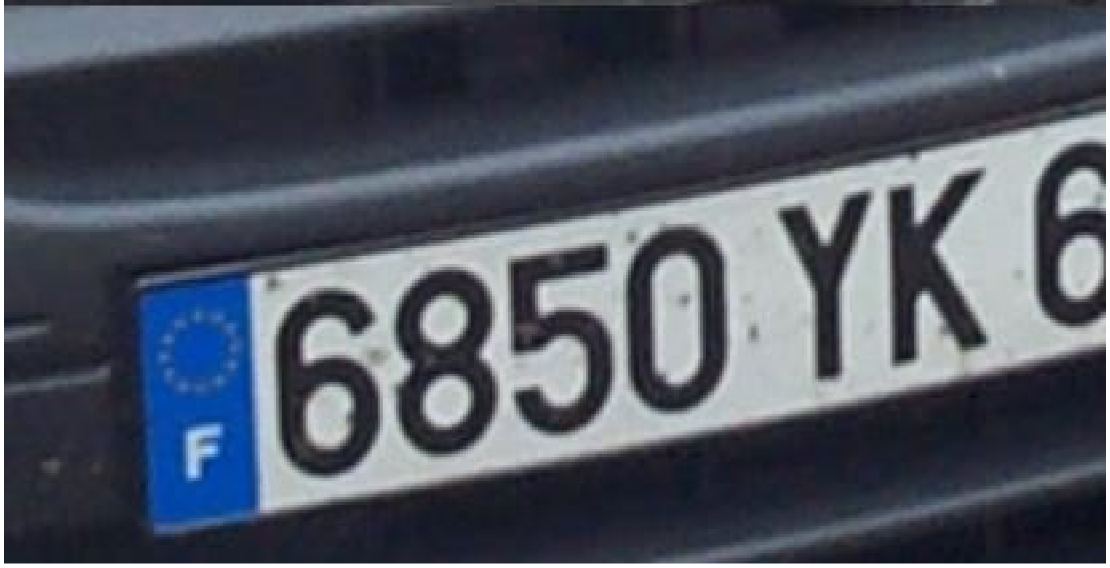
```
This is a lot of 12 point text to test the
ocr code and see if it works on all types
of file format.

The quick brown dog jumped over the
lazy fox. The quick brown dog jumped
over the lazy fox. The quick brown dog
jumped over the lazy fox. The quick
brown dog jumped over the lazy fox.
```

```python
img2 =
cv2.cvtColor(io.imread('https://b2633864.smushcdn.com/2633864/wp-
content/uploads/2017/06/example_01.png'), cv2.COLOR_BGR2RGB)
cv2_imshow(img2)

# Adding custom options
custom_config = r'--oem 3 --psm 6'
print(pytesseract.image_to_string(img2, config=custom_config))
```

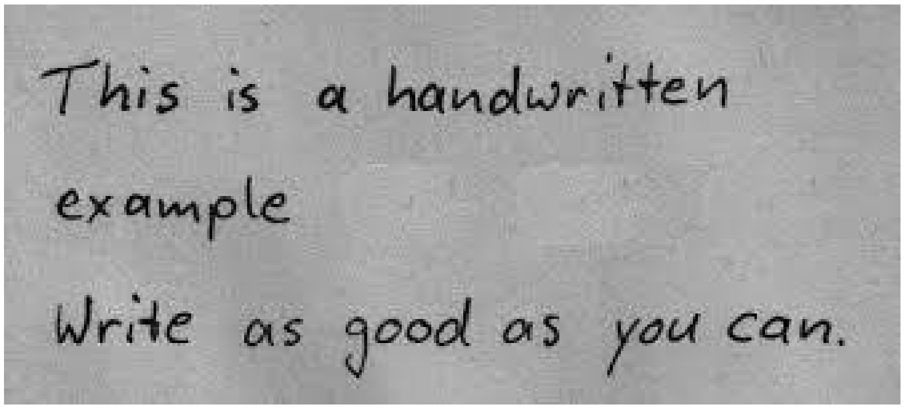


```
Noisyimage
to test
Tesseract OCR


img3 =
cv2.cvtColor(io.imread('https://upload.wikimedia.org/wikipedia/commons
/0/06/FrenchNumberPlates.jpg'), cv2.COLOR_BGR2RGB)
cv2_imshow(img3)

# Adding custom options
custom_config = r'--oem 3 --psm 6'
print(pytesseract.image_to_string(img3, config=custom_config))
```



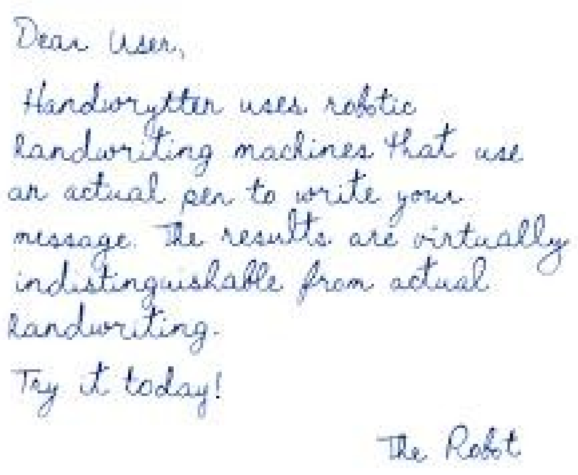


```
ae


img4 = cv2.cvtColor(io.imread('/content/download.jpg'),
cv2.COLOR_BGR2RGB)
cv2_imshow(img4)

# Adding custom options
custom_config = r'--oem 3 --psm 6'
print(pytesseract.image_to_string(img4, config=custom_config))
```



This is a handwritten
example
Write as good as you can.

```
This is a handwritten
example
Write as geoal as you can.


img5 = cv2.cvtColor(io.imread('/content/234.png'), cv2.COLOR_BGR2RGB)
cv2_imshow(img5)

# Adding custom options
custom_config = r'--oem 3 --psm 6'
print(pytesseract.image_to_string(img5, config=custom_config))
```



Dear User,
Handwrytten uses robotic
handwriting machines that use
an actual pen to write your
message. The results are virtually
indistinguishable from actual
handwriting.
Try it today!
The Robot

```
Dean Wn,
slanlioryptin, roketio
tendon ing medina Hal wn
an attial pen te yorile gous
sissoge ta salle wad rua
indifingualat ln al
k ing
Tey today!
```

```
“te, Robt
```