

# Optical Character Recognition (OCR) using OpenCV and Tesseract in Python

**Sungkyeong (Lucia) Jeon**

2020 WAY (Winter At Yonsei)

AI & Design

Prof. Jin Kook Lee

# Contents

- 1) Topic
- 2) Hand-written Data
- 3) OpenCV and pytesseract
- 4) Results
- 5) References

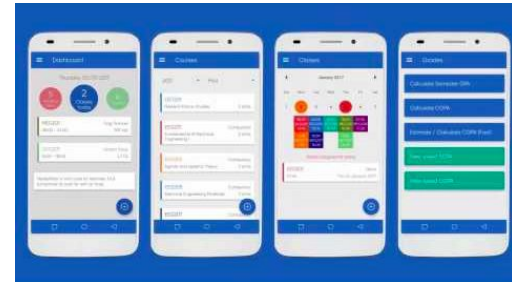
# Topic: Study-planner App Development

## General Scheme

- compares two schedules: before-plan, after-plan
- plan input using hand-written data (specially targeted for iPad users)
- shows the efficiency as a percentage (i.e. You achieved 75% of the plans today!)
- store the personalized data (how long can this person concentrate, how many different subjects worked best, personalized tips for study efficiency improvement)
- modules needed: time management module / app UI module / calculation (AID) module
- further improvement ideas
  - Google calendar-like interface
  - add a timer functionality for each plan so that the user can take advantage of it

## For this project,

- implementation of AID module
- takes in start\_time, end\_time as float type and file\_name as str type
- use text recognition (OCR) to convert image file to a string
- calculate and evaluate the efficiency of the day



# Hand-written Data for Testing

Own handwritten notes from iPad GoodNotes

Lossless join decomposition.

How to achieve? /  $R_1 : \alpha \cup \beta, PK: \alpha$   
 $R_2 : R - (\beta - \alpha)$

LHS Superkey

keep records of

Exam 2

lecture 7: Deterministic Selection

Spring 2021

OS 4

Software Eng 3

ON DELETE CASCADE

ON DELETE SET NULL

ON DELETE SET ID=1

lecture 13

lecture 14 : dynamic pt 1

lecture 15 : dynamic pt 2

abortive

↳ abort, abortion

abridge

↳ abridge, abridged, abridgment

Implementing

Nonlinear Equations (ch. 5)

absolute

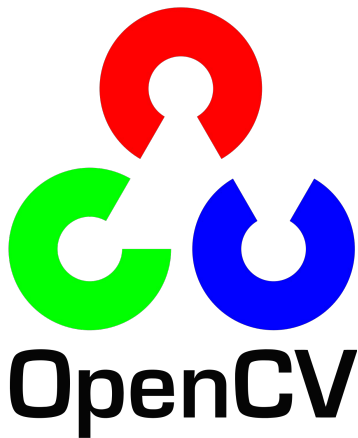
absolve ≠ abdicate

→ Intermediate Value Theorem

$\text{sign}(f(a)) \neq \text{sign}(f(b))$

functions can have any number of solutions

# OpenCV and Pytesseract



- Open source computer vision
- supports python library (import cv2)
- imread(filename), cvtColor(),  
imshow(file\_directory) etc



Tesseract OCR

- developed by Google since 2006
- contains a command line program
- supports more than 100 languages
- Pytesseract builds on top of Tesseract

```
# Lucia Jeon  
# AID Final Project  
# 01/17/2021
```

```
import os  
import cv2  
import pytesseract  
from pytesseract import image_to_string
```

```
class Plan:
```

```
    def __init__(self, start_time, end_time, name):  
        # start_time (float), end_time(float), name(str)  
        self.start_time = start_time  
        self.end_time = end_time  
        self.name = name
```

```
plan_count = int(input("How many plans do you have?: "))  
plan_list = list()
```

```
for i in range(1, plan_count+1):  
    start_time = float(input(str(i)+"- start time: "))  
    end_time = float(input(str(i)+"- end time: "))  
    file_name = input(str(i)+"- image file name: ")  
    img=cv2.imread(file_name)  
    name = image_to_string(img)  
    print(name)
```

```
# create a Schedule class instance
```

```
plan_list.append(Plan(start_time, end_time, name))
```

```
# evaluate each plans
```

```
plan_efficiency = list()
```

```
for i in range(plan_count):
```

```
    actual_start_time = float(input(str(i+1)+"- actual start time: "))
```

```
    actual_end_time = float(input(str(i+1)+"- actual end time: "))
```

```
    efficiency = (actual_end_time - actual_start_time) / (plan_list[i].e
```

```
    print("plan", i+1, "efficiency:", efficiency, "%")
```

```
    plan_efficiency.append(efficiency)
```

```
print("average efficiency:", sum(plan_efficiency)/len(plan_efficiency))
```

Source code snippets

# Results

- 1) Download opencv-python on terminal using Homebrew

```
[Lucias-MacBook-Air-219:~ luciajeon$ pip3 install opencv-python
Collecting opencv-python
  Downloading opencv_python-4.5.1.48-cp39-cp39-macosx_10_13_x86_64.whl (40.3 MB)
    |████████████████████████████████████████| 40.3 MB 87 kB/s
Collecting numpy>=1.19.3
  Downloading numpy-1.19.5-cp39-cp39-macosx_10_9_x86_64.whl (15.6 MB)
    |████████████████████████████████████████| 15.6 MB 15.0 MB/s
Installing collected packages: numpy, opencv-python
Successfully installed numpy-1.19.5 opencv-python-4.5.1.48
```

- 2) Download pytesseract on terminal using pip3 command

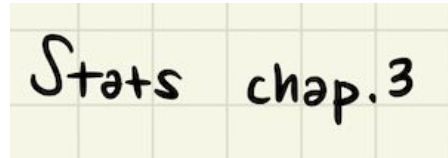
```
[Lucias-MacBook-Air-219:~ luciajeon$ pip3 install pytesseract
Collecting pytesseract
  Using cached pytesseract-0.3.7.tar.gz (13 kB)
Collecting Pillow
  Downloading Pillow-8.1.0-cp39-cp39-macosx_10_10_x86_64.whl (2.2 MB)
    |████████████████████████████████████████| 2.2 MB 1.6 MB/s
Using legacy 'setup.py install' for pytesseract, since package 'wheel' is not installed.
Installing collected packages: Pillow, pytesseract
  Running setup.py install for pytesseract ... done
Successfully installed Pillow-8.1.0 pytesseract-0.3.7
```

cf. both modules can be installed by either Homebrew or pip command

# Results

```
==== RESTART: /Users/luciajeon/Desktop/proj
How many plans do you have?: 2
1- start time: 9
1- end time: 12
1- image file name: plan1.png
Stats chap.3
2- start time: 15
2- end time: 18
2- image file name: plan2.png
ML textbook pg.100-140
1- actual start time: 9
1- actual end time: 11.5
plan 1 efficiency: 83.33333333333334 %
2- actual start time: 15.5
2- actual end time: 17.5
plan 2 efficiency: 66.66666666666666 %
average efficiency: 75.0
```

plan1.png



plan2.png





# References

- pyimagesearch “Using Tesseract OCR with Python”  
(<https://www.pyimagesearch.com/2017/07/10/using-tesseract-ocr-python/>)
- StackOverflow: TesseractNotFoundError resolve,
- Homebrew: OpenCV and Tesseract Library Install (<https://formulae.brew.sh/formula/tesseract>)
- Tessdoc (Tesseract documentation) on Github  
(<https://tesseract-ocr.github.io/tessdoc/Installation.html>)
- Illinois Library “Introduction to OCR and Searchable PDFs”  
(<https://guides.library.illinois.edu/c.php?g=347520&p=4121425>)