IAV Assignment-1 160122771080

5.Implement Optical Character Recognition (OCR) for Handwritten Text

```python
import cv2
import numpy as np
import pytesseract
import easyocr
import os

# Set Tesseract OCR path (Update this based on your system)
pytesseract.pytesseract.tesseract_cmd = r"/usr/bin/tesseract"

def preprocess_image(image_path):
    """Load and preprocess the image for better OCR accuracy."""

    # Check if image file exists
    if not os.path.exists(image_path):
        raise FileNotFoundError(f"Error: Image file not found at {image_path}")

    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

    # Ensure image is loaded correctly
    if image is None:
        raise ValueError(f"Error: Unable to load the image. Check the file format and path: {image_path}")

    # Apply thresholding to enhance text
    _, thresh = cv2.threshold(image, 127, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)

    # Denoise the image
    denoised = cv2.fastNlMeansDenoising(thresh, h=30)

    return denoised

def extract_text_tesseract(image):
    """Extract text using Tesseract OCR."""
    custom_config = r'--oem 3 --psm 6'
    text = pytesseract.image_to_string(image, config=custom_config)
    return text.strip()

def extract_text_easyocr(image_path):
    """Extract text using EasyOCR."""
    reader = easyocr.Reader(['en'])  # Initialize EasyOCR for English
    text = reader.readtext(image_path, detail=0)
    return " ".join(text)

def main():
    """Main function to run OCR."""
    # Provide correct image file name (update path if needed)
    image_path = r"/content/image1.jpg"

    print("\nPreprocessing Image...")
    processed_image = preprocess_image(image_path)

    # Save processed image (for debugging)
    cv2.imwrite("processed_image.jpg", processed_image)

    print("\nExtracting Text with Tesseract OCR...")
    tesseract_text = extract_text_tesseract(processed_image)
    print("Tesseract OCR Output:\n", tesseract_text)

    print("\nExtracting Text with EasyOCR...")
    easyocr_text = extract_text_easyocr(image_path)
    print("EasyOCR Output:\n", easyocr_text)

if __name__ == "__main__":
    main()

```

```
reprocessing Image...

xtracting Text with Tesseract OCR...
ARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU.
ARNING:easyocr.easyocr:Downloading detection model, please wait. This may take several minutes depending upon your network connectio
esseract OCR Output:
STARTING
OUR OWN
USINESS
SN'T JUST
```

```
. JOB -
T'S A WAY
F LIFE.

xtracting Text with EasyOCR...
rogress: |██████████████████████████████████████████████████| 100.0% CompleteWARNING:easyocr.easyocr:Downloading recognition model,
rogress: |██████████████████████████████████████████████████| 100.0% CompleteEasyOCR Output:
Starting Your own business isn't just A job - it'sa way Of Life. Richard Branson
```

```
!pip install pytesseract
```

```
Requirement already satisfied: pytesseract in /usr/local/lib/python3.11/dist-packages (0.3.13)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.11/dist-packages (from pytesseract) (24.2)
Requirement already satisfied: Pillow>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from pytesseract) (11.1.0)
```

```
 !sudo apt install tesseract-ocr
```

```
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 29 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 1s (8,974 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/Front
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 ... 124947 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) ...
```

another code

```python
import cv2
import pytesseract
def ocr_core(img):
    text = pytesseract.image_to_string(img)
    return text
img = cv2.imread('/content/image1.jpg')
def get_grayscale(image):
    return cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
def remove_noise(image):
    return cv2.medianBlur(image,5)
def thresholding(image):
    return cv2.threshold(image, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)[1]
    cv2.waitKey(0)

img = get_grayscale(img)
img = thresholding(img)
img = remove_noise(img)
print(ocr_core(img))

```

```
STARTING
YOUR OWN
BUSINESS
ISN'T JUST

A J06-
IT'S A WAY
```

```
OF LIFE.
```

```
import numpy as np
import cv2

# Open webcam
webcam = cv2.VideoCapture(0)

if not webcam.isOpened():
    print("Error: Could not open webcam")
    exit()

# Define color ranges
colors = {
    "Red": ([136, 87, 111], [180, 255, 255], (0, 0, 255)),
    "Green": ([25, 52, 72], [102, 255, 255], (0, 255, 0)),
    "Blue": ([94, 80, 2], [120, 255, 255], (255, 0, 0))
}

while True:
    ret, imageFrame = webcam.read()
    if not ret:
        print("Error: Could not read frame")
        break

    hsvFrame = cv2.cvtColor(imageFrame, cv2.COLOR_BGR2HSV)
    kernel = np.ones((5, 5), "uint8")

    for color_name, (lower, upper, color) in colors.items():
        lower, upper = np.array(lower, np.uint8), np.array(upper, np.uint8)
        mask = cv2.inRange(hsvFrame, lower, upper)
        mask = cv2.dilate(mask, kernel)
        contours, _ = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

        for contour in contours:
            area = cv2.contourArea(contour)
            if area > 300:
                x, y, w, h = cv2.boundingRect(contour)
                cv2.rectangle(imageFrame, (x, y), (x + w, y + h), color, 2)
                cv2.putText(imageFrame, f"{color_name} Color", (x, y - 10),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.6, color, 2)

    cv2.imshow("Multiple Color Detection", imageFrame)

    # Press 'q' to exit
    if cv2.waitKey(10) & 0xFF == ord('q'):
        break

webcam.release()
cv2.destroyAllWindows()
```

```
Error: Could not open webcam
Error: Could not read frame
```

Data structures for Image Analysis -Write a program that computes the T-pyramid of an image

```

import cv2
import matplotlib.pyplot as plt
# Import the necessary function
from google.colab.patches import cv2_imshow

# Load the image
img = cv2.imread('/content/image1.jpg')
layer=img.copy()
for i in range(4):
  plt.subplot(2,2,i+1)
  layer=cv2.pyrDown(layer)
  plt.imshow(layer)
  # Use cv2_imshow instead of cv2.imshow
  cv2_imshow(layer)
  cv2.waitKey(0)

cv2.destroyAllWindows()
```

STARTING
YOUR OWN
BUSINESS
ISN'T JUST
A JOB –
IT'S A WAY
OF LIFE.

Richard Branson

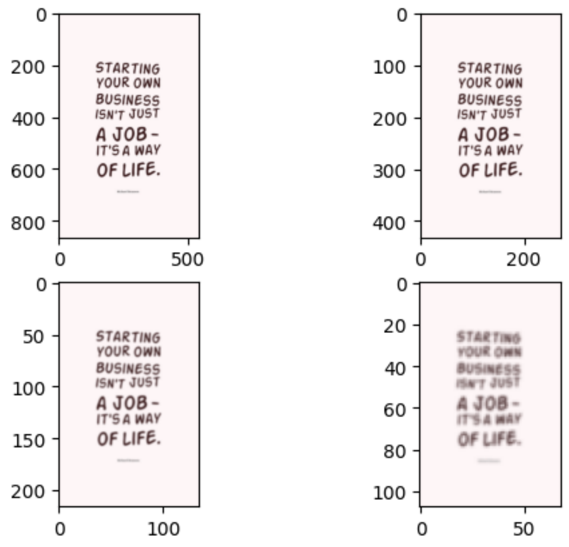
STARTING
YOUR OWN
BUSINESS
ISN'T JUST
A JOB –
IT'S A WAY
OF LIFE.

Richard Branson

STARTING
YOUR OWN
BUSINESS
ISN'T JUST
A JOB –
IT'S A WAY
OF LIFE.

Richard Branson

STARTING
YOUR OWN
BUSINESS
ISN'T JUST



Sample project for Image Smoothing

```
import cv2
import numpy as np
from google.colab.patches import cv2_imshow

image =cv2.imread('/content/image1.jpg')
kernel2=np.ones((5,5),np.float32)/25
img=cv2.filter2D(src=image,ddepth=-1,kernel=kernel2)
cv2_imshow(image)
cv2_imshow(img)
cv2.waitKey()
cv2.destroyAllWindows()
```

STARTING YOUR OWN BUSINESS ISN'T JUST A JOB – IT'S A WAY OF LIFE.

Richard Branson

STARTING
YOUR OWN
BUSINESS
ISN'T JUST
A JOB –
IT'S A WAY
OF LIFE.

Richard Branson

Sample project for Edge detection using Sobel ,Canny edge.

```python
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt
import os

# Provide the correct path to the image file
# Removed extra space from the path
img_path = '/content/image1.jpg'  # Update this path if needed

# Check if the file exists before reading it
if not os.path.exists(img_path):
    raise FileNotFoundError(f"Error: Image file not found at {img_path}")

# Read the image in grayscale
img = cv.imread(img_path, cv.IMREAD_GRAYSCALE)
assert img is not None, "File could not be read, check with os.path.exists()"

# Compute Sobel gradients
sobel_x = cv.Sobel(img, cv.CV_64F, 1, 0, ksize=5)
sobel_y = cv.Sobel(img, cv.CV_64F, 0, 1, ksize=5)

# Compute gradient magnitude (combined Sobel)
sobel_combined = np.sqrt(sobel_x**2 + sobel_y**2)
sobel_combined = cv.convertScaleAbs(sobel_combined)  # Convert to uint8

# Canny Edge Detection
edges = cv.Canny(img, 100, 200)

# Plot the results
plt.figure(figsize=(12, 8))

plt.subplot(2, 2, 1)
plt.title('Original Image')
plt.imshow(img, cmap='gray')
```