Imports and Configuration

```python
import cv2
import pytesseract
```

- `cv2` **(OpenCV):** Used for image processing tasks like loading, converting to grayscale, thresholding, and saving images.
- `pytesseract` **(Tesseract-OCR):** A Python wrapper for Google's Tesseract-OCR, which extracts text from images.

```python
pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
```

- This line **sets the path** to the Tesseract-OCR executable on Windows.
- This is **needed only for Windows users**, as `pytesseract` doesn't automatically find Tesseract unless it's added to the system's PATH.

Processing Each Image

```python
for image_path in image_files:
```

- **Loops through each file** in the `image_files` list.

```python
    print(f"PROCESSING {image_path}...")
```

- Prints which image is currently being processed.

Loading the Image

```python
image = cv2.imread(image_path)
```

- **Loads the image** from the given file path using OpenCV.
- If the file doesn't exist or there is an issue, `cv2.imread()` returns `None`.

```python
if image is None:
    print(f"❌ ERROR: Could not load {image_path}. Skipping.")
    continue
```

- **Checks if the image loaded successfully.**
- If `image is None`, it **prints an error message a↓kips** to the next image.

Convert Image to Grayscale

```python
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

- **Converts the image to grayscale** (black & white shades).
- This improves OCR accuracy, as colors can interfere with text recognition.

Apply Thresholding (Binarization)

```python
processed_img = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)[1]
```

- **Thresholding converts the image into pure black and white pixels** (binarization).
- `cv2.THRESH_BINARY + cv2.THRESH_OTSU`:
  - **OTSU's method** automatically determines the best threshold value.
  - Improves text clarity for OCR by removing shadows, noise, or gradients.

Extract Text Using Tesseract OCR

python Copy Edit

```
text = pytesseract.image_to_string(processed_img)
```

- Passes the **processed image** to Tesseract OCR for **text extraction**.
- The extracted text is stored in the `text` variable.

Display Extracted Text

python Copy Edit

```
print(f"📜 EXTRACTED TEXT FROM {image_path}:\n{text}\n")
```

- Prints the extracted text from the image.

Save the Processed Image

python Copy Edit

```
processed_file = f"processed_ {image_path}"
```

- **Creates a new filename** by adding `"processed_"` as a prefix to the original filename.

```python
cv2.imwrite(processed_file, processed_img)
```

- **Saves the processed (thresholded) image** to disk for debugging or review.

```python
print(f"✅ PROCESSED IMAGE SAVED AS {processed_file}\n")
```

- Prints confirmation that the processed image was saved.

Display the Processed Image

```python
cv2.imshow(f"PROCESSED - {image_path}", p..essed_img)
```

- **Displays the processed image** in a new window.

```python
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- `cv2.waitKey(0)` waits for a key press before closing the window.
- `cv2.destroyAllWindows()` closes all OpenCV windows.

Bug in the Code

```python
cv2.destroyAllWindows() EXPALIN LINE BY LINE
```