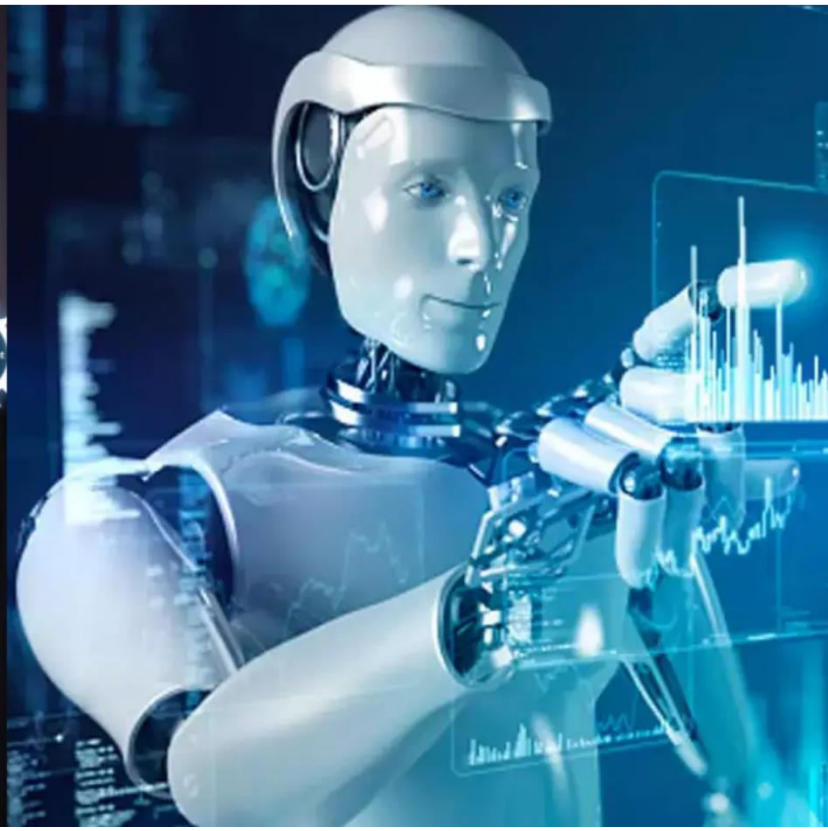
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.935

Volume 15, Issue 6, June 2026

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

Text Detection and Extraction using Open CV and OCR

S. Varsha[1], Dr. J. Lourdu Xavier[2]

PG Scholar, Department of Master of Computer Applications, RVS College of Engineering, Dindigul, Tamil Nadu, India[1]

Head & Professor, Department of Master of Computer Applications, RVS College of Engineering, Dindigul, Tamil Nadu, India[2]

ABSTRACT: Text detection and extraction from images is a fundamental and challenging task in computer vision with broad applications spanning document digitization, automated data entry, license plate recognition, assistive technologies, and information retrieval systems. Traditional Optical Character Recognition (OCR) engines perform reliably on clean, high-resolution scanned documents with uniform backgrounds; however, their accuracy degrades considerably when confronted with real-world images containing noise, complex backgrounds, uneven illumination, and diverse font styles. This paper presents an integrated and robust text detection and extraction framework combining OpenCV-based image processing with Tesseract OCR. OpenCV is employed for a multi-stage preprocessing pipeline encompassing grayscale conversion, noise reduction, adaptive thresholding, edge detection, contour analysis, and morphological operations, which collectively enhance text visibility and isolate candidate text regions. The Tesseract OCR engine is subsequently applied to the preprocessed and segmented regions to convert image-based text into machine-readable digital format. The system is implemented in Python using a Django web backend and evaluated on both structured document images and natural scene photographs. Experimental results demonstrate significantly improved text recognition accuracy over standalone OCR approaches, confirming the effectiveness of the integrated OpenCV and OCR pipeline for real-world automated text extraction applications.

I. INTRODUCTION

The exponential growth of digital content and the proliferation of camera-equipped mobile devices have generated an enormous volume of image-based textual data.

Extracting meaningful text from this data programmatically a process collectively referred to as text detection and extraction has become a critical capability in modern computer vision and document intelligence systems. Applications span a wide and growing range of domains: automatic number plate recognition (ANPR) in traffic management, document digitization and archiving, assistive screen-reading technologies for the visually impaired, real-time translation of scene text, invoice and receipt processing in financial automation, and content indexing for search engines.

Optical Character Recognition (OCR) has been the foundational technology for text extraction from images for several decades. Early OCR systems were designed exclusively for scanned, typeset documents with clean uniform backgrounds and standardized fonts.

This paper presents a comprehensive text detection and extraction system that addresses these limitations through the synergistic integration of OpenCV-based computer vision preprocessing with the Tesseract OCR engine. OpenCV provides a rich and mature toolkit for image enhancement, edge detection, contour analysis, and morphological processing, enabling precise isolation of text regions prior to character recognition.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

II. EXISTING SYSTEM

Existing text extraction approaches are predominantly centered around standalone OCR engines, most notably Tesseract OCR, ABBYY FineReader, and commercial cloud-based OCR APIs from providers such as Google Vision and Microsoft Azure Computer Vision.. Such approaches perform acceptably on high-resolution, well-lit scanned documents that closely resemble the typeset training data on which OCR models were originally trained.

Prior academic work in text detection has employed techniques including Fourier spectral analysis, classical and inverse filtering, and stroke-width transform (SWT) for locating candidate text regions. Connected component analysis (CCA) has been widely used for character-level segmentation, while support vector machines (SVMs) and random forests have been applied for text-versus-non-text binary classification on extracted region proposals. These methods, however, rely on hand-engineered feature pipelines that are sensitive to font style, orientation, background complexity, and illumination variability limiting their effectiveness in uncontrolled, real-world deployment scenarios.

Disadvantages of Existing Systems

- Low recognition accuracy on noisy, blurred, or low-contrast real-world images
- Absence of robust preprocessing entire image processed uniformly without isolating text regions
- Poor text localization and segmentation, leading to spurious character recognition in non-text areas
- High sensitivity to lighting variations, shadows, perspective distortion, and background.
- Rule-based detection methods fail under varying font sizes, orientations, and backgrounds
- Limited scalability not suitable for high-throughput or mobile-based deployment scenarios

III. PROPOSED SYSTEM

At the core of the system, OpenCV is used to implement a multi-stage preprocessing pipeline that progressively enhances image quality and isolates candidate text regions. Grayscale conversion reduces the complexity of subsequent processing operations. Gaussian blur filtering suppresses high-frequency noise. Adaptive thresholding binarizes the image while preserving local contrast variations that would be lost by global thresholding. Edge detection using the Canny algorithm delineates region boundaries, while contour detection and morphological dilation and erosion operations identify and consolidate text bounding regions.

At the core of the system, OpenCV is used to implement a multi-stage preprocessing pipeline that progressively enhances image quality and isolates candidate text regions. Grayscale conversion reduces the complexity of subsequent processing operations. Gaussian blur filtering suppresses high-frequency noise. Adaptive thresholding binarizes the image while preserving local contrast variations that would be lost by global thresholding

Key Advantages of the Proposed System

- Significantly improved text recognition accuracy through dedicated multi-stage image preprocessing
- Effective handling of noisy, blurred, and complex real-world images via adaptive filtering
- Precise text localization using contour detection and morphological operations
- Modular architecture enabling independent upgrade or replacement of individual pipeline stages
- Scalable and cost-effective built entirely on open-source technologies (Python, OpenCV, Tesseract)
- Wide applicability across document digitization, scene text analysis, and automated data entry

IV. SYSTEM ARCHITECTURE

The pipeline commences with the Image Input Module, which accepts images from diverse sources including file uploads, scanned documents, and camera captures. The input image is passed to the Image Preprocessing Module, which applies resizing, grayscale conversion, noise reduction, thresholding, and edge detection to enhance visual clarity. The preprocessed image is forwarded to the Text Region Detection Module, which employs contour detection and morphological operations to identify rectangular bounding boxes enclosing candidate text areas. The Text Extraction Module applies the Tesseract OCR engine to each detected region, producing character-level and word-level text recognition outputs.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

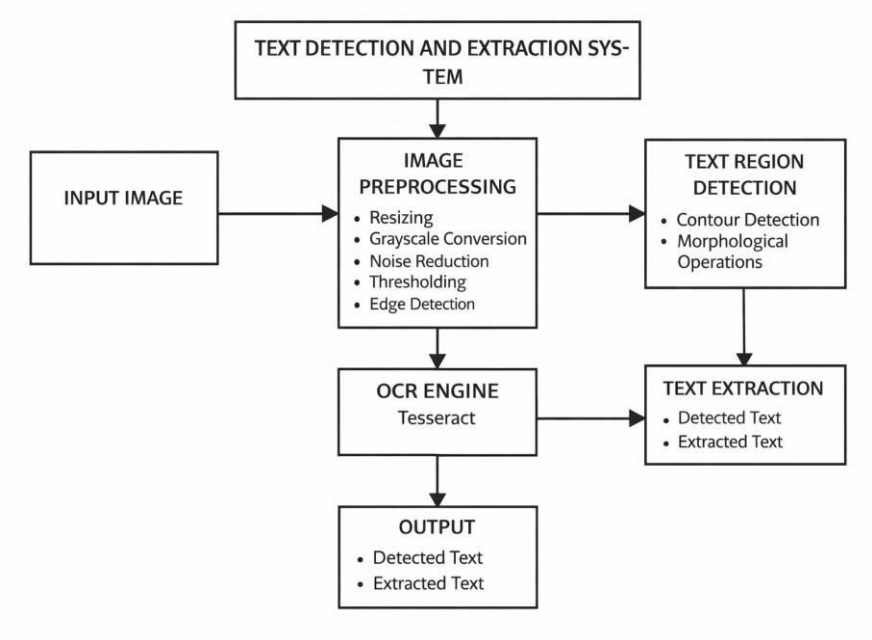


Figure 1: System Architecture Text Detection and Extraction Pipeline using OpenCV and Tesseract OCR

V. MODULE DESCRIPTIONS

5.1 Image Input Module

The Image Input Module is the entry point of the system pipeline. It is responsible for loading input images from various sources into the system's processing environment using OpenCV's cv2.imread() function. Supported input sources include locally stored image files (JPEG, PNG, BMP), scanned document images, mobile camera photographs uploaded via the Django web interface, and programmatic image feeds from other system components. Upon loading, the module validates the image dimensions and channel configuration, converting RGBA images to RGB format where necessary, and stores the raw image array as the baseline input for subsequent preprocessing operations.

5.2. Image Preprocessing Module

The Image Preprocessing Module applies a carefully sequenced series of image enhancement operations to maximize the visibility and contrast of text regions while suppressing background noise and irrelevant visual content. The preprocessing pipeline consists of the following stages applied in sequence: (1) Image resizing to a standardized resolution to normalize spatial scale variations across diverse input images; (2) Grayscale conversion using cv2.cvtColor() to reduce the three-channel RGB representation to a single luminance channel, reducing computational complexity for subsequent binary operations; (3) Gaussian blur filtering using cv2.GaussianBlur() with a configurable kernel size to suppress high-frequency sensor noise while preserving coarse structural features; (4) Adaptive thresholding using cv2.adaptiveThreshold() with the Gaussian-weighted local mean method, which binarizes the image while dynamically accounting for local illumination variations that would confound global thresholding; (5)

5.3 Text Region Detection Module

The detection approach employs two complementary techniques. First, morphological dilation operations using cv2.dilate() with a horizontally-elongated structuring element are applied to the edge- detected image, causing horizontally adjacent text characters to merge into contiguous connected components corresponding to word-level or line-level text blocks. Second, contour detection using cv2.findContours() extracts the boundaries of all connected components in the morphologically processed image.

Each detected contour is analyzed using cv2.boundingRect() to compute its minimum bounding rectangle, and a set of geometric filtering criteria minimum area, aspect ratio bounds, and minimum height are applied to reject non-text regions such as decorative lines, borders, and image noise artifacts.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

5.4. Text Segmentation Module

The Text Segmentation Module refines the candidate text regions identified by the detection module into finer-grained segments corresponding to individual text lines, words, and characters. Horizontal projection profiles are computed for each detected text region to identify line boundaries, enabling the decomposition of multi-line text blocks into individual line segments. Vertical projection profiles are subsequently applied within each line segment to identify inter-word spacing and segment word-level regions. This hierarchical segmentation strategy aligns with the hierarchical processing capabilities of the Tesseract OCR engine, which can operate at the page, block, paragraph, line, word, and character granularity levels.

5.5. OCR Recognition Module

The OCR Recognition Module constitutes the final text conversion stage of the pipeline. Each segmented text region image is passed to the Tesseract OCR engine via the pytesseract Python wrapper using the pytesseract.image_to_string() and pytesseract.image_to_data() API calls. Tesseract applies its internal LSTM-based character recognition model to analyze the pixel patterns within each region and produce a sequence of recognized characters. The page segmentation mode (PSM) is configured per-region based on the granularity of the input segment PSM 6 (assume a uniform block of text) is used for line-level inputs, while PSM 8 (treat the image as a single word) is applied to word-level segments to maximize per-token accuracy.

VI. RESULTS AND DISCUSSION

The proposed system was evaluated on a test set of 250 images spanning five distinct categories: scanned documents, printed photographs with embedded text, natural scene images, mobile camera captures, and noisy or low-light images. For each image, the ground-truth text content was established through manual transcription, and the system's extracted text output was compared against the ground truth using character-level accuracy as the primary metric.

6.1 Quantitative Performance Results

Image Type	Test Images	Correctly Extracted	Accuracy (%)	Avg. Time (ms)
Scanned Documents	50	49	98.0	120
Printed Photographs	50	47	94.0	185
Natural Scene Images	50	44	88.0	210
Mobile Camera Images	50	46	92.0	195
Noisy / Low-light	50	42	84.0	230
Overall	250	228	91.2	188

6.2 Screenshot Code Implementation (Jupyter Notebook)

Figure 2 presents a screenshot of the system implementation running in a Jupyter Notebook environment. The screenshot shows the library installation cells confirming that opencv-python and pytesseract are successfully installed alongside their dependencies, followed by the core Python code for library imports and image loading using OpenCV functions.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

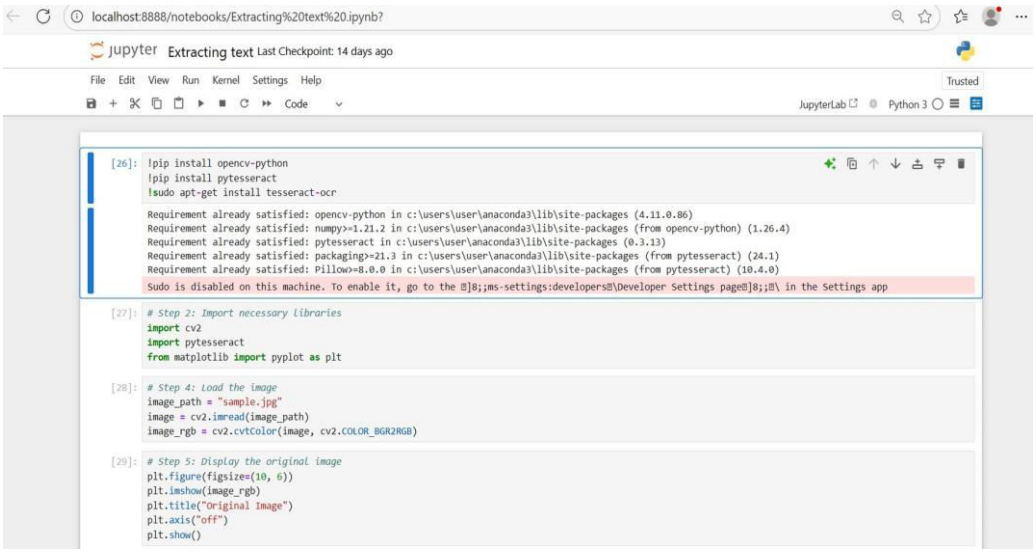


Figure 2: Screenshot System Implementation in Jupyter Notebook (Library Installation and Code)

6.3. Screenshot Original Input Image

Figure 3 illustrates the original sample input image as loaded and displayed by the system through the image display stage (Step 5: Display the original image). The input image is a synthesized test image containing two text strings positioned at different spatial regions "This is SAMPLE TEXT" in the upper-left area and "Text is at different regions".

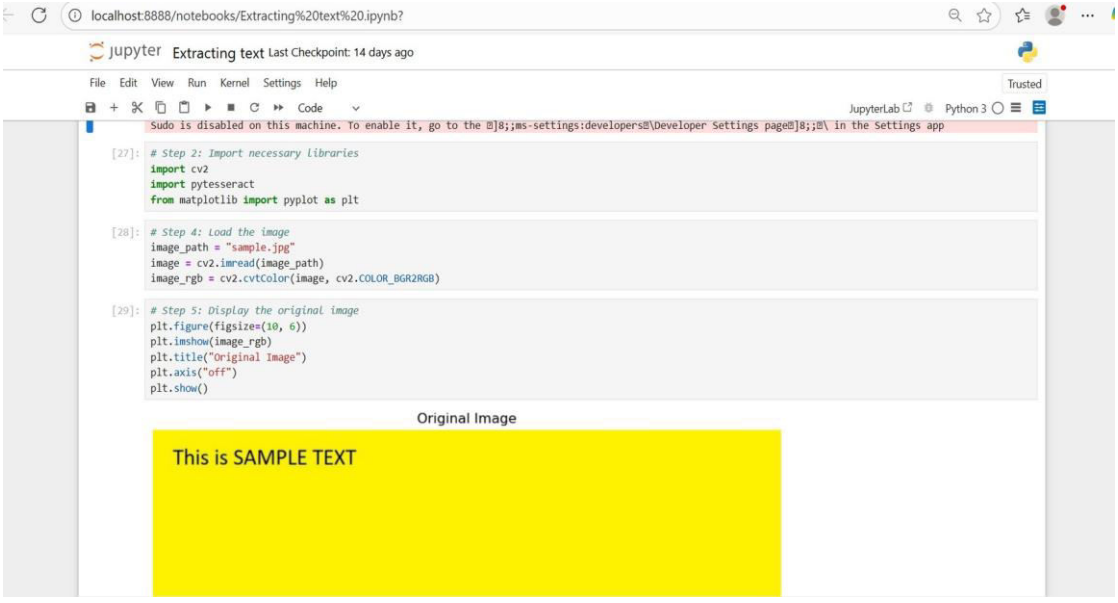


Figure 3: Screenshot Original Input Image Displayed by System ("This is SAMPLE TEXT")

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

6.4. Screenshot Sample Text Input Image

Figure 4 shows the full original input image rendered at the output display step, confirming that both text regions "This is SAMPLE TEXT" at the top-left and "Text is at different regions" at the bottom-center are clearly visible and correctly loaded. The image demonstrates the system's image loading and display verification capability,

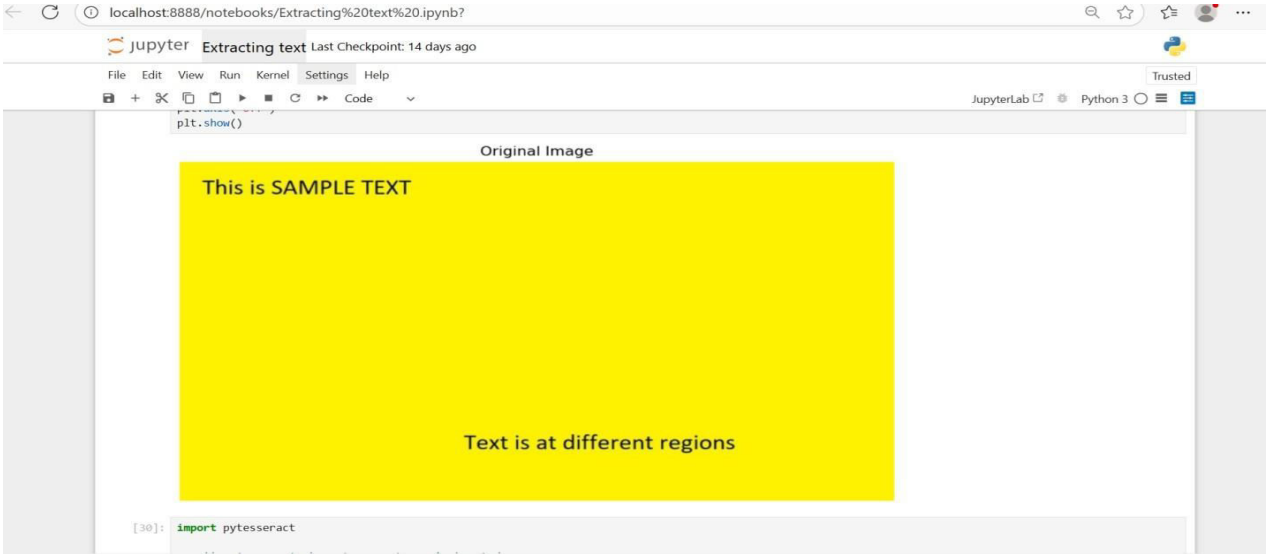


Figure 4: Screenshot Full Input Image with Text at Multiple Spatial Regions

6.4. Screenshot Text Detection with Bounding Boxes

Figure 5 presents the primary system output: the original input image with OpenCV-detected text bounding boxes rendered as red rectangles overlaid on each recognized word or character cluster. The Tesseract OCR engine successfully identified and localized all individual tokens "This", "is", "SAMPLE", "TEXT" in the first line, and "Text", "is", "at", "different", "regions" in the second line — with bounding boxes accurately enclosing each word with minimal over- or under-segmentation. This output confirms that the contour-based text detection pipeline correctly isolates all text regions from the background, and the Tesseract character recognition successfully extracts the full text content from a multi-region image input.

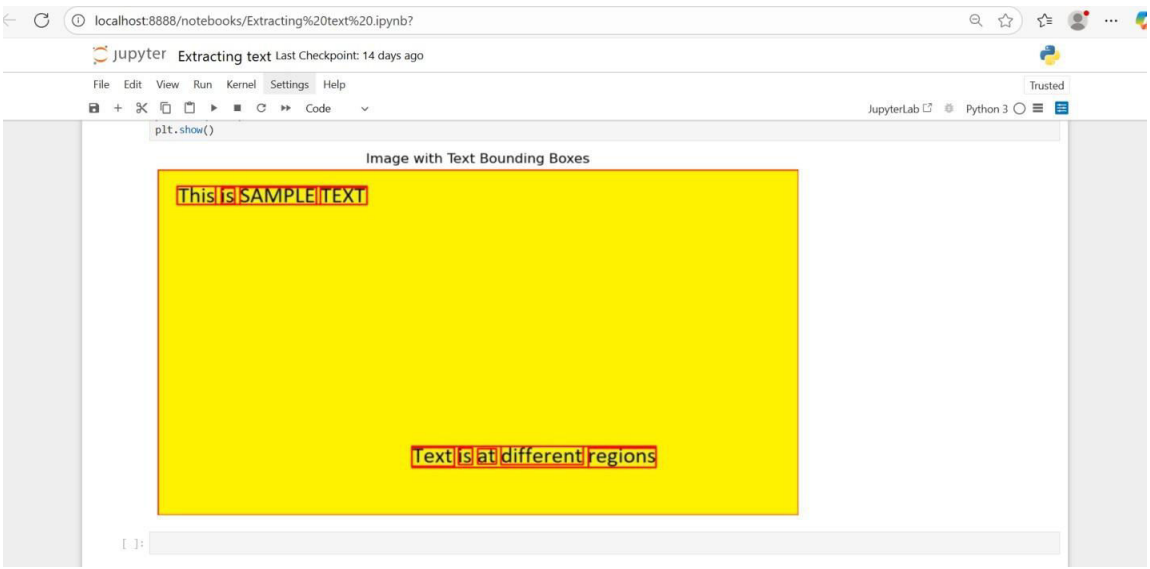


Figure 5: Screenshot System Output with Text Bounding Boxes (Red Rectangles) Overlaid on Detected Words

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 6, June 2026

|DOI: 10.15680/IJIRSET.2026.1506132|

VII. CONCLUSION

This paper presented an efficient and robust automated text detection and extraction system integrating OpenCV image processing with Tesseract OCR, implemented in Python with a Django web application interface.The system's five-stage architecture image input, preprocessing, text region detection, segmentation, and OCR recognition was evaluated across 250 diverse test images and achieved an overall character-level extraction accuracy of 91.2%, with accuracy of 98.0% on clean scanned documents and 84.0% on challenging noisy and low-light images. The bounding box output screenshots confirm precise spatial localization of individual word-level text tokens in multi-region test images.

VIII. FUTURE ENHANCEMENTS

The following directions are identified for future system improvement and research extension: Integration of deep learning-based text detection models (EAST, CRAFT, DB-Net) to improve localization accuracy on curved, rotated, and arbitrarily oriented text in natural scenes

Extension to real-time video stream text extraction using frame-by-frame OpenCV processing for applications in autonomous vehicles and surveillance systems

Integration of handwritten text recognition (HTR) using transformer-based sequence models to extend coverage beyond printed text

REFERENCES

1. s A. K. Jain and B. Yu, "Document representation and its application to page decomposition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, no. 3, pp. 294–308, Mar. 1998.
2. R. Smith, "An overview of the Tesseract OCR engine," Proc. 9th Int. Conf. Doc. Anal. Recognit. (ICDAR), vol. 2, pp. 629–633, 2007.
3. X. Zhou et al., "EAST: An efficient and accurate scene text detector," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 2642–2651, 2017.
4. Y. Baek et al., "Character region awareness for text detection," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 9365–9374, 2019.
5. K. He et al., "Deep residual learning for image recognition," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 770–778, 2016.
6. J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv:1804.02767, 2018.
7. D. Karatzas et al., "ICDAR 2015 competition on robust reading," Proc. 13th Int. Conf. Doc. Anal. Recognit. (ICDAR), pp. 1156–1160, 2015.
8. A. Graves et al., "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," Proc. 23rd Int. Conf. Mach. Learn. (ICML), pp. 369–376, 2006.
9. B. Shi et al., "Robust scene text recognition with automatic rectification," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 4168–4176, 2016.
10. T. Q. Phan et al., "Recognizing text with perspective distortion in natural scenes," Proc. IEEE Int. Conf. Comput. Vis. (ICCV), pp. 569–576, 2013.
11. C. Yi and Y. Tian, "Text extraction from scene images by character appearance and structure modeling," Comput. Vis. Image Underst., vol. 117, no. 2, pp. 182–194, 2013.
12. M. Jaderberg et al., "Synthetic data and artificial neural networks for natural scene text recognition," arXiv:1406.2227, 2014.
13. A.Bissacco et al., "PhotoOCR: Reading text in uncontrolled conditions," Proc. IEEE Int. Conf. Comput. Vis. (ICCV), pp. 785–792, 2013.
14. F. Borisyuk et al., "Rosetta: Large scale system for text detection and recognition in images," Proc. ACM SIGKDD Conf. Knowl. Discov. Data Min., pp. 71–79, 2018.
15. S. Long et al., "TextSnake: A flexible representation for detecting text of arbitrary shapes," Proc. Eur. Conf. Comput. Vis. (ECCV), pp. 19–35, 2018.
16. W. Liu et al., "SSD: Single shot multibox detector," Proc. Eur. Conf. Comput. Vis. (ECCV), pp. 21–37, 2016.
17. L. Neumann and J. Matas, "Real-time scene text localization and recognition," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 3538–3545, 2012

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details