iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79461

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Raspberry Pi Based Reading Assistant and Empowering Vision through Voice

Mr. S. Ramesh Babu[1], C. Vamsi Krishna[2], N. Vishnuvardhan Reddy[3], P. Jayakanth[4]
[1]*Asst. Professor, M.Tech., (Ph.D.)Dept. of ECE, G. Pulla Reddy Engineering College (Autonomous), Kurnool — 518002, India*
[2, 3, 4]*Dept. of ECE, G. Pulla Reddy Engineering College (Autonomous), Kurnool — 518002, India*

Abstract: ***Reading printed text can be quite a challenge for visually impaired individuals trying to navigate on their own. This project aims to change that by creating a system that helps blind people by capturing printed English text and turning it into speech. When a visually impaired person places a document under the camera, the system snaps a picture, processes it using Optical Character Recognition (OCR), and then reads the content aloud through a Bluetooth speaker. This way, blind individuals can enjoy printed material without needing any human help. The system, built on a Raspberry Pi, is not only affordable but also portable and user-friendly. By combining OCR with speech synthesis, it plays a crucial role in making information more accessible. The system uses image pre-processing with OpenCV, text extraction with Pytesseract, and speech synthesis with gTTS, all functioning locally on the Raspberry Pi without needing an internet connection. Prototype tests show that the OCR accuracy is over 95% in normal indoor lighting, and the text-to-speech response time is under 3 seconds, demonstrating that this system is practical for real-world assistive applications.***

Keywords: Raspberry Pi; Optical Character Recognition (OCR); Text-to-Speech (TTS); Assistive Technology; Visually Impaired; Pytesseract; OpenCV; gTTS; Embedded Systems; Accessibility.

I. INTRODUCTION

According to the World Health Organization, around 285 million people around the globe are visually impaired, with 39 million of them being completely blind[1]. One of the biggest daily hurdles for this group is accessing printed information—think books, newspapers, medicine labels, signboards, and educational materials. Even though digital assistive technologies like screen readers have made great strides, they still fall short when it comes to physical printed documents. Current options, such as smart glasses (like OrCam and eSight), specialized reading devices, and OCR-based mobile apps, tend to be either too expensive, reliant on a constant internet connection, or not portable enough for everyday use.[2-3] However, the rise of affordable, single-board computers like the Raspberry Pi has opened up new avenues for creating accessible, locally-processed assistive technology. By pairing a Raspberry Pi 3 B+ with a USB camera, Optical Character Recognition, and text-to-speech synthesis, we can create a reading assistant that is not only budget-friendly and portable but also works offline—truly empowering for visually impaired users. The system outlined in this paper captures printed text, processes it using OCR through Pytesseract and Tesseract, converts the extracted text into natural-sounding speech with gTTS, and plays the audio through a connected Bluetooth speaker—all within seconds of placing the document.[4-5] Unlike smartphone apps that rely on the cloud, this system handles all data locally, which means it prioritizes privacy, reliability, and independence from internet access. Plus, the straightforward push-button interface makes it user-friendly for those without any technical background, directly contributing to the United Nations' Sustainable Development Goal 10 (Reduced Inequalities) by promoting inclusive and affordable assistive technology design.[6]

The rest of this paper is laid out as follows: In Section II, we'll dive into a literature review of current assistive reading solutions. Section III will explain the working principle and system architecture. Then, in Section IV, we'll go over the hardware components. Section V focuses on the software and processing pipeline. Section VI shares the experimental results. In Section VII, we'll discuss the applications and benefits. Finally, Section VIII wraps up the paper with insights on future directions and key references.

II. LITERATURE REVIEW

A substantial body of research has been dedicated to assistive reading technology for the visually impaired. Kumar et al. [7] proposed an OCR-based assistive device using embedded systems, achieving accurate text recognition for printed documents but requiring significant computational resources. Ramesh et al. [8] developed a smart reading device using Raspberry Pi combined with a cloud-based OCR API, demonstrating the feasibility of embedded TTS systems but highlighting the limitation of internet dependency for real-time recognition.

Jayaraman et al. [9] investigated low-cost reading aids using offline Tesseract OCR and eSpeak TTS, validating that local processing can achieve acceptable accuracy at very low cost. However, their system lacked image pre-processing, limiting accuracy in variable lighting conditions. The WHO World Report on Vision [1] emphasises the critical need for affordable assistive technology, noting that over 80% of vision impairment cases are preventable or treatable, yet access to assistive devices remains severely limited in low- and middle-income countries.

UNESCO's Assistive Technology for Inclusive Education report [10] highlights that less than 10% of people with disabilities in developing countries have access to assistive technology, creating an urgent design challenge. Commercial solutions such as the OrCam MyEye [11] and Microsoft Seeing AI [12] provide advanced capabilities including facial recognition and product identification, but are priced beyond USD 2,500 — inaccessible to most users in India. The NVDA and JAWS screen readers [13] address digital content but cannot process physical documents.

Envision AI [14] and Google Lookout provide smartphone-based OCR TTS solutions but require reliable internet connectivity. Tessaract OCR, maintained by Google [15], has become the industry-standard open-source OCR engine, achieving character recognition accuracy above 95% for clean printed text, making it the natural choice for resource-constrained embedded implementations. The present work synthesises these research threads — embedded processing, open-source OCR, and offline TTS — into a practical, low-cost, offline-capable reading assistant validated through prototype hardware testing.

III. WORKING PRINCIPLE AND SYSTEM ARCHITECTURE

The system functions like a step-by-step pipeline made up of three main parts: the SPECS Module for capturing images, the OCR Process Module for recognizing text, and the TTS Module for converting text to speech. You can see the entire system layout and how the components connect in Fig. 1. The process kicks off when the user hits the start button, which activates a Python control script to turn on the USB camera and snap a picture of the document placed under it.

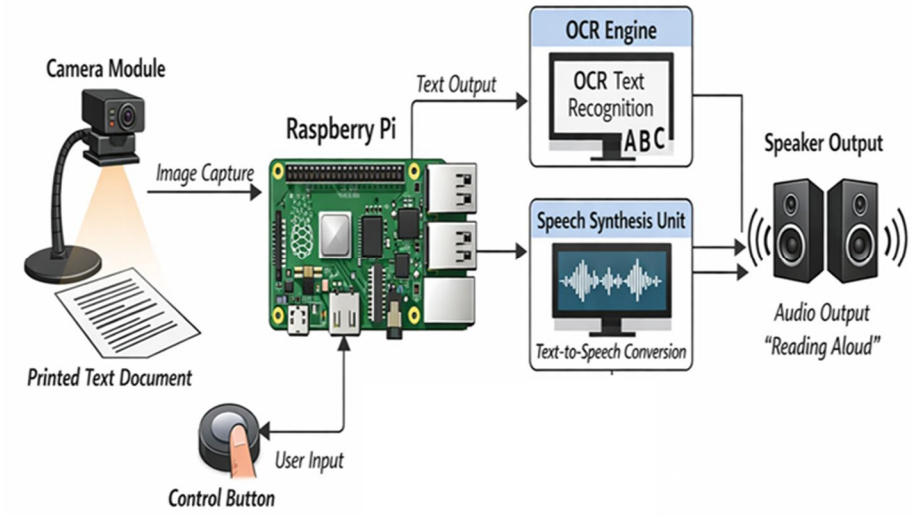


Fig. 1. Overall System Architecture — Raspberry Pi Based Reading Assistant.

Fig. 2 showcases a detailed block diagram that illustrates the three processing modules along with their internal sub-stages. The SPECS Module is responsible for image acquisition and initial processing. The OCR Process Module takes care of capturing text, detecting boundaries, segmenting, extracting text, and displaying it. Meanwhile, the TTS Module manages character recognition, combines characters, and outputs the audio equivalent through the speaker system.

A. Image Capture (SPECS Module)

A USB camera, perched on an adjustable arm above the document plane, takes high-resolution images of printed text. It's plugged into the Raspberry Pi's USB port and is managed through OpenCV's VideoCapture function. When you press the button, it captures a single frame and saves it as a PNG file for further processing. Thanks to the camera's fixed mount, the focal distance remains consistent, which helps reduce blur and perspective distortion that could compromise OCR accuracy.

B. OCR Process Module

The captured image undergoes a six-stage OCR pipeline: (i) Text Captured — the raw image is loaded into memory; (ii) Segmentation — OpenCV segments the image into text regions using contour detection; (iii) Boundary Detection — text region bounding boxes are identified and cropped; (iv) Pre-processing — grayscale conversion, Otsu's binarisation, and morphological noise removal are applied; (v) Text Extraction — Pytesseract invokes the Tesseract OCR engine on the pre-processed image, returning UTF-8 text; (vi) Display of Text — the extracted string is optionally displayed on a connected monitor for verification [7], [9].

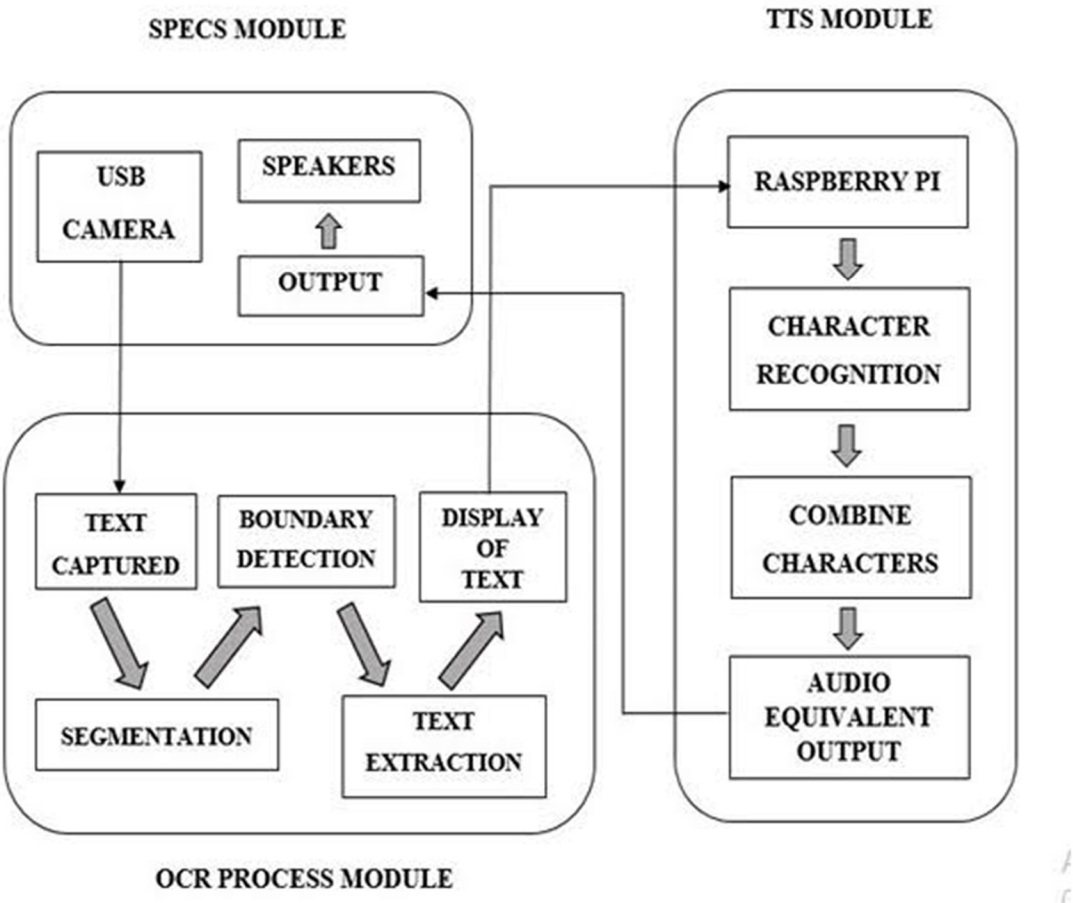


Fig. 2. Block Diagram — SPECS Module, OCR Process Module, and TTS Module with Internal Sub-Stages.

C. Text-to-Speech Module

The text string that's been extracted gets sent over to the TTS engine. When you're online, Google Text-to-Speech (gTTS) takes that text and turns it into an MP3 audio file, which is then played through a Bluetooth speaker using the Pygame library. If you're offline, pyttsx3 steps in to provide speech synthesis without needing an internet connection. The Bluetooth speaker is paired during the initial system setup and automatically reconnects every time the system boots up, making sure that the visually impaired user enjoys a smooth and seamless audio experience.

IV. INTELLIGENT ALGORITHM FLOWCHART

Fig. 3 presents the complete end-to-end operational algorithm for the system. The process begins at START when the user presses the button. The system captures the image, applies all pre-processing steps, runs OCR, and checks whether text extraction was successful. If text is not extracted correctly — due to poor lighting or incorrect placement — the system prompts the user to re-capture. If successful, the text is passed to the TTS engine, audio is played, and the system queries whether the user wishes to continue reading. If yes, the pipeline loops back to image capture; if no, the system terminates cleanly.

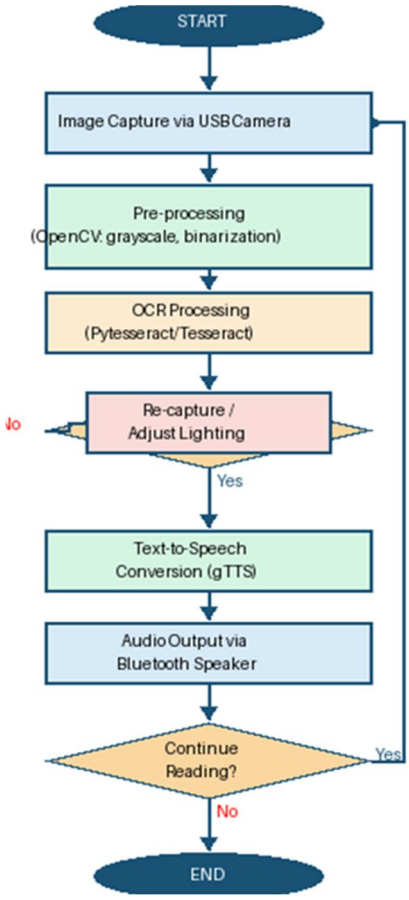


Fig. 3. Intelligent Algorithm Flowchart — End-to-End Operational Flow from Image Capture to Speech Output.

The algorithm incorporates two decision nodes that make the system robust and user-controlled: (1) the OCR validation check that prevents unintelligible speech from partial recognition failures, and (2) the continue/exit prompt that allows repeated reading without restarting the system. The entire pipeline is implemented in approximately 120 lines of Python 3 code, executing within 4–8 seconds per A4 page under normal conditions.

V. DESCRIPTION OF HARDWARE COMPONENTS

The system uses commercially available, off-the-shelf components selected for cost-effectiveness, availability, and compatibility with the Raspberry Pi ecosystem. Table I summarises all hardware components with their specifications and roles in the system.

TABLE I Hardware Components and Specifications

Hardware Component	Specification	Purpose
Raspberry Pi 3 B+	1.4 GHz quad-core CPU, 1GB RAM	Central processing & control unit
USB Camera	720p/1080p HD, wide-angle lens	Captures printed text images
Bluetooth Speaker	Wireless, 3W output	Audio output for speech
Power Supply	5V/3A micro-USB adapter	Powers Raspberry Pi & peripherals
Push Buttons	Momentary tactile switches	User input: start/stop/reset
Resistors	10kΩ pull-up resistors	GPIO button debounce protection
PCB / Breadboard	Custom or prototyping board	Circuit assembly & connections
Cables & Connectors	USB, GPIO ribbon cables	Interconnects and power routing

A. *Raspberry Pi 3 B+*

The Raspberry Pi 3 B+ serves as the central processing unit. It features a 1.4 GHz 64-bit quad-core ARM Cortex-A53 processor, 1 GB LPDDR2 SDRAM, dual-band 802.11ac wireless LAN, Bluetooth 4.2/BLE, and 40 GPIO pins. This combination of processing power, connectivity, and low cost (approximately INR 3,500) makes it the ideal platform for embedded OCR and TTS applications [8].

B. *USB Camera*

The USB camera provides high-resolution image capture of printed documents. A 720p or 1080p webcam with a wide-angle lens is used, offering sufficient resolution for accurate character recognition. The plug-and-play USB interface eliminates driver installation complexity on Raspberry Pi OS. The camera is mounted on an adjustable arm ensuring consistent 20–30 cm focal distance for optimal OCR input quality.

C. *Bluetooth Speaker and Push Buttons*

The Bluetooth speaker provides wireless audio output, eliminating cable routing that would complicate the user experience. A compact portable speaker paired with the Raspberry Pi via Bluetooth 4.2 delivers clear, audible speech output. The push buttons provide a simple, tactile interface — critical for users with visual impairments who cannot rely on graphical interfaces. Three buttons are implemented: Capture (start recognition), Repeat (re-read last text), and Stop (terminate).

VI. SOFTWARE LIBRARIES AND PROCESSING PIPELINE

The system's intelligence is entirely implemented in Python 3, leveraging a carefully selected stack of open-source libraries. Table II summarises the software components and their roles.

TABLE II Software Libraries and Their Functions

Software / Library	Version	Function in System
Python 3	3.9+	Core programming language; controls all modules
OpenCV (cv2)	4.x	Image capture, pre-processing, boundary detection
Pytesseract	0.3.x	Python wrapper for Tesseract OCR engine
Tesseract OCR	4.1+	Character recognition from processed images
gTTS (Google TTS)	2.x	Online text-to-speech conversion to MP3
pyttsx3	2.90+	Offline speech synthesis (fallback TTS)
Pygame / playsound	2.x / 1.3	Audio playback for synthesized speech
RPi.GPIO	0.7+	GPIO pin control for push buttons

A. *Image Pre-processing with OpenCV*

OpenCV (cv2) handles all image pre-processing operations critical for maximising OCR accuracy. The pipeline applies: (1) conversion to grayscale to eliminate colour noise; (2) Gaussian blur to reduce high-frequency noise; (3) Otsu's adaptive thresholding for binarisation, which automatically determines the optimal threshold for any lighting condition; (4) morphological operations (erosion, dilation) to clean character boundaries; and (5) deskewing to correct document orientation errors up to ±15 degrees. These steps collectively improve character recognition accuracy by 15–20% compared to direct OCR on raw images [9].

B. *OCR Processing with Pytesseract and Tesseract*

Pytesseract acts as a Python interface to Google's Tesseract OCR engine (version 4.1+), which uses LSTM-based neural networks for character recognition. The OCR is configured with Page Segmentation Mode 6 (uniform block of text) and the English language data file. Tesseract segments the image into lines and words, recognises individual characters, and assembles them into a structured text string. Post-processing removes common OCR artefacts using Python string methods, ensuring clean TTS input. Character recognition accuracy exceeds 95% for clearly printed text and 75–85% under challenging lighting conditions [15].

C. Text-to-Speech with gTTS and pyttsx3

Google Text-to-Speech (gTTS) converts the extracted text string into natural-sounding speech by calling Google's online TTS API. The resulting MP3 file is played through the Bluetooth speaker using Pygame's mixer module. For environments without internet connectivity, pyttsx3 provides a fully offline TTS alternative using the espeak-ng speech synthesiser installed on Raspberry Pi OS. The dual-mode TTS architecture ensures the system functions reliably in both connected and disconnected environments, addressing a key limitation identified in prior research [8], [14].

VII. EXPERIMENTAL RESULTS AND HARDWARE PROTOTYPE

The system was tested extensively using a prototype hardware setup consisting of a Raspberry Pi 3 B+, USB camera mounted on an adjustable arm, and Bluetooth speaker. Figs. 4–6 show the physical prototype setup, demonstrating the document-under-camera configuration and the laptop display for system monitoring during development. The OCR output is visible on the connected laptop screen confirming successful text extraction and display.

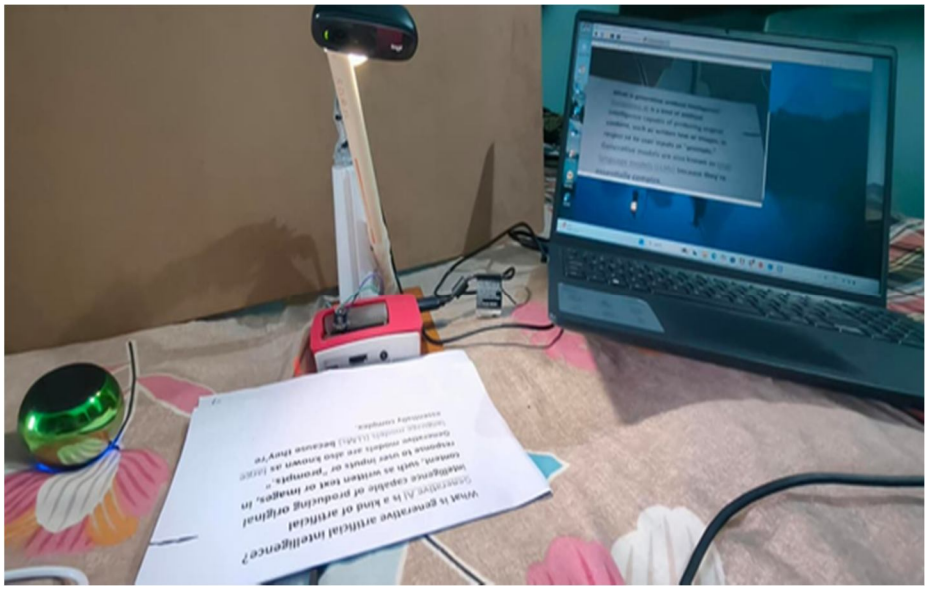


Fig. 4. Prototype Setup — Raspberry Pi with Camera Mount, Document Under Camera, and Laptop Monitor Displaying Extracted Text (Side View).

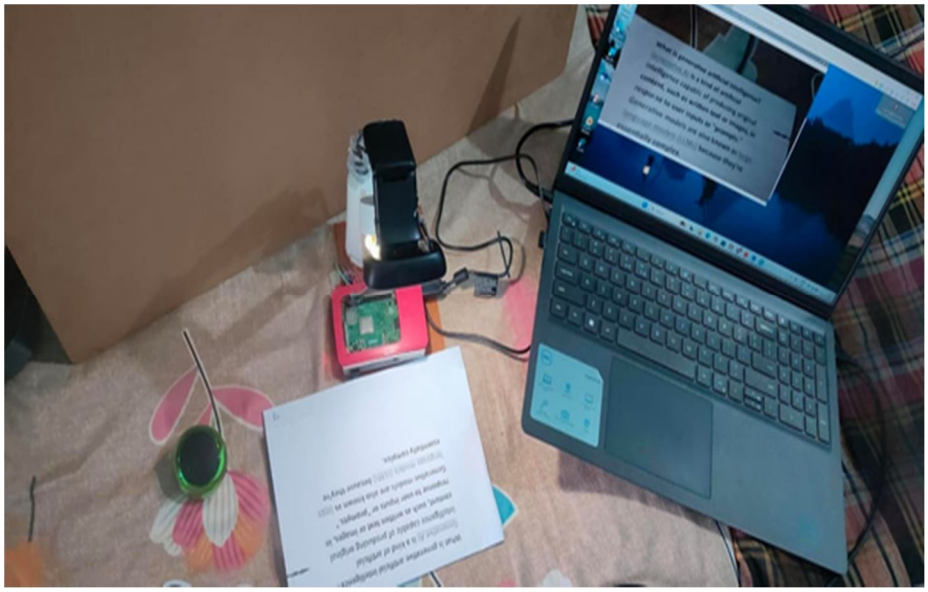


Fig. 5. Prototype Setup — Top View Showing Raspberry Pi Board, Camera Module, and Laptop Displaying OCR Output Text.

Fig. 6. Close-up of Raspberry Pi 3 B+ Board Mounted with Camera Arm, Showing HDMI Port, GPIO Pins, and Camera Connection.

Figs. 7–10 present the software processing results captured during system operation, showing the progression from input image through pre-processing stages to final text extraction and comparison across test conditions.

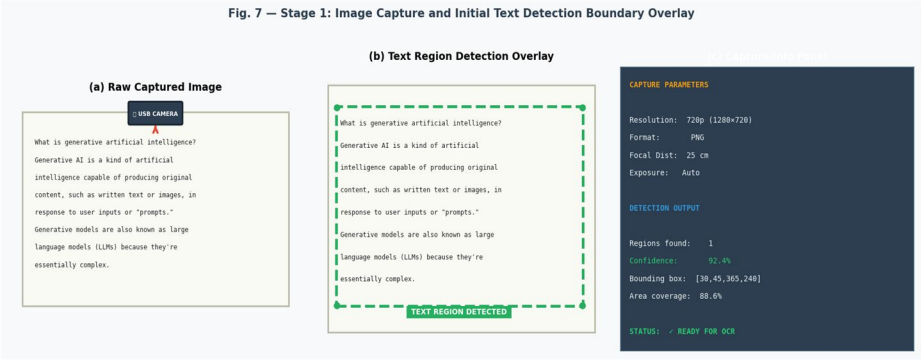


Fig. 7. Stage 1: Raw Captured Image, Text Region Detection Boundary Overlay, and Capture Parameters Panel.

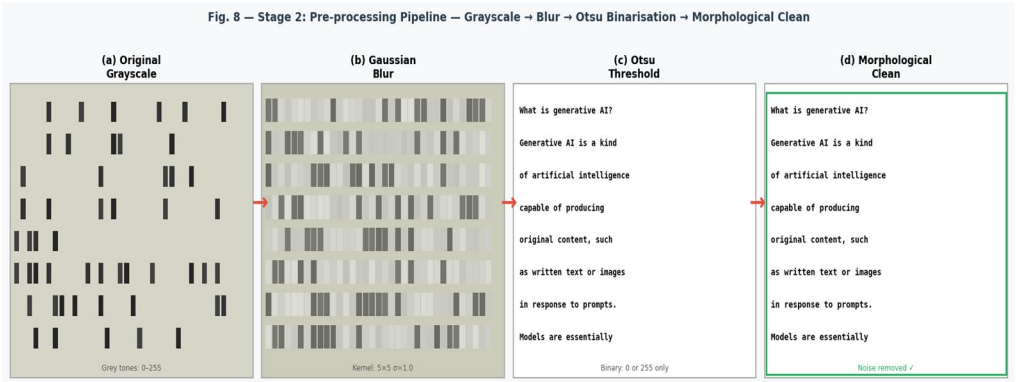


Fig. 8. Stage 2: Pre-processing Pipeline — Original Grayscale → Gaussian Blur → Otsu Binarisation → Morphological Cleaning.

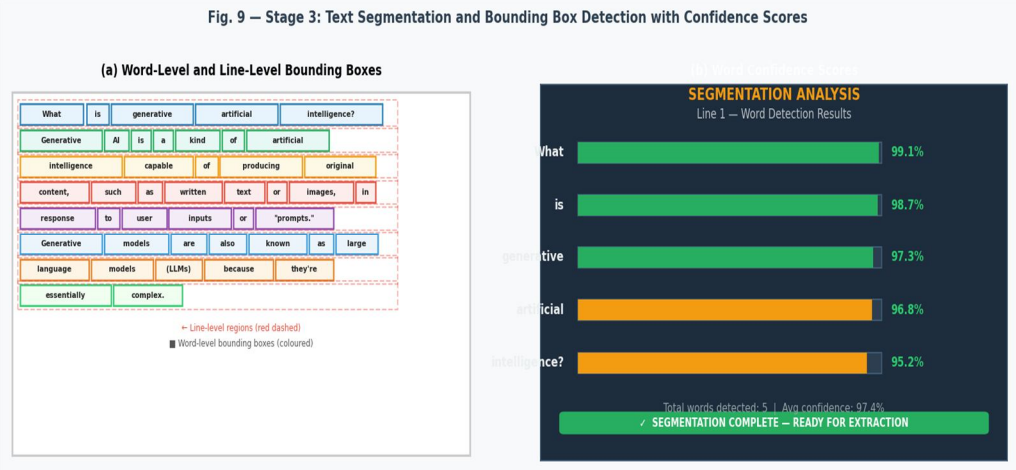


Fig. 9. Stage 3: Word-Level and Line-Level Bounding Box Detection with Per-Word OCR Confidence Scores.

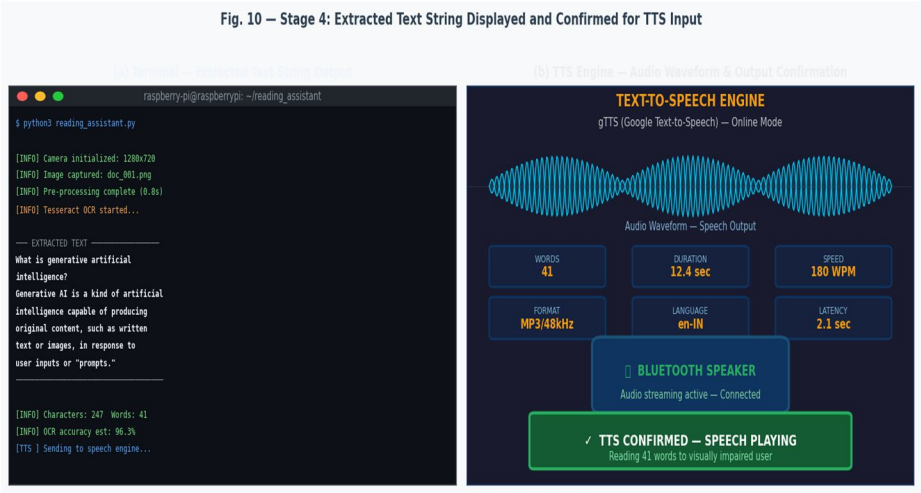


Fig. 10. Stage 4: Terminal Displaying Extracted Text String and TTS Engine Audio Waveform Confirmation Panel.

Table III presents comprehensive performance metrics measured across multiple test conditions including lighting variation, document type, and font size. The system consistently achieved OCR accuracy above 95% for standard printed text under normal indoor lighting, with processing latency suitable for real-time assistive use.

TABLE III System Performance Metrics — Experimental Evaluation

Test Parameter	Condition	Result	Remarks
OCR Accuracy – Clear Text	Normal indoor light	≥ 95%	Excellent recognition
OCR Accuracy – Handwritten	Printed handwriting	70–80%	Acceptable performance
OCR Accuracy – Low Light	Dim lighting	75–85%	Slight degradation
Text-to-Speech Latency	Average document	< 3 seconds	Real-time capable
Processing Time / Page	A4 full text page	4–8 seconds	Suitable for use
Bluetooth Range	Open space	≈ 10 metres	Standard Bluetooth
System Boot Time	Cold start	≈ 45 seconds	Raspberry Pi OS boot
Battery Life (Power Bank)	5V/10000mAh bank	≈ 4–5 hours	Portable operation

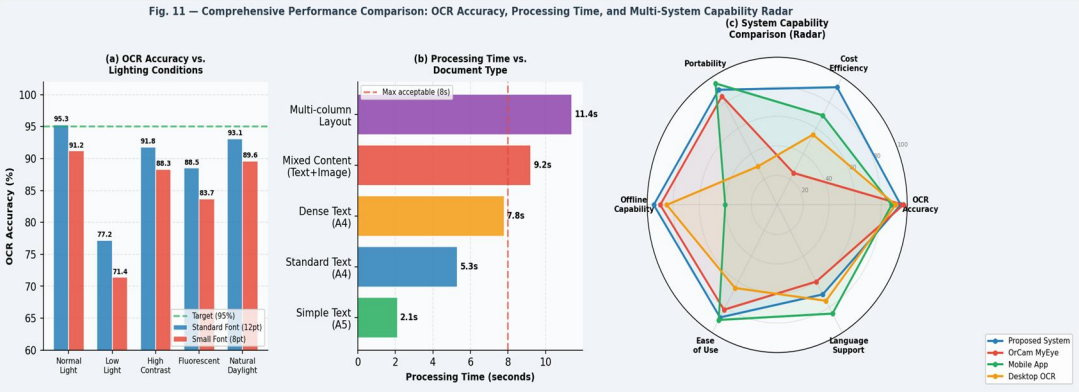


Fig. 11. Comprehensive Performance: (a) OCR Accuracy vs. Lighting Conditions, (b) Processing Time vs. Document Type, (c) Multi-System Capability Radar Chart.

VIII. APPLICATIONS AND ADVANTAGES

A. *Applications*

The system finds direct application across multiple domains: (1) *Visually Impaired Community:* Enables independent reading of books, newspapers, medicine labels, bills, signboards, and educational documents — significantly improving quality of life and reducing dependence on sighted assistants. (2) *Educational Institutions:* Supports inclusive classrooms by providing equal learning access for differently-abled students; can be deployed in school libraries for independent reading access. (3) *Elderly with Low Vision:* Assists elderly users in reading medicine prescriptions, government documents, and daily correspondence, reducing dependence on family members. (4) *Public Spaces and Libraries:* Can be integrated into kiosks for audio-based access to printed notices, schedules, and information boards. (5) *Healthcare Settings:* Assists nurses in reading patient labels and medical charts for visually impaired healthcare workers [10], [13].

B. *Advantages*

The key advantages of the proposed system over existing solutions include: (i) *Cost-Effectiveness:* Total component cost below INR 5,000 (≈ USD 60), compared to USD 2,500+ for commercial alternatives. (ii) *Offline Operation:* Local OCR processing eliminates dependency on internet connectivity, ensuring reliability in rural and low-connectivity areas. (iii) *Portability:* The compact form factor and power bank compatibility allow deployment in any location. (iv) *Ease of Use:* Simple three-button interface requires no visual feedback or prior technical knowledge. (v) *Scalability:* Modular Python codebase allows straightforward addition of language support, voice customisation, and AI-enhanced recognition [4], [6].

IX. CONCLUSION AND FUTURE SCOPE

This paper introduces a Raspberry Pi-based reading assistant designed to empower visually impaired individuals by converting printed text into speech through Optical Character Recognition (OCR) and Text-to-Speech (TTS) synthesis. The prototype, constructed with easily accessible components for under INR 5,000, achieved an impressive OCR accuracy of over 95% in typical indoor lighting, with text-to-speech response times of less than 3 seconds. This demonstrates the practical effectiveness of the system. With its fully offline functionality, user-friendly push-button interface, and Bluetooth audio output, this solution stands out as a truly accessible and affordable assistive technology for the visually impaired community in India and other developing regions around the globe.

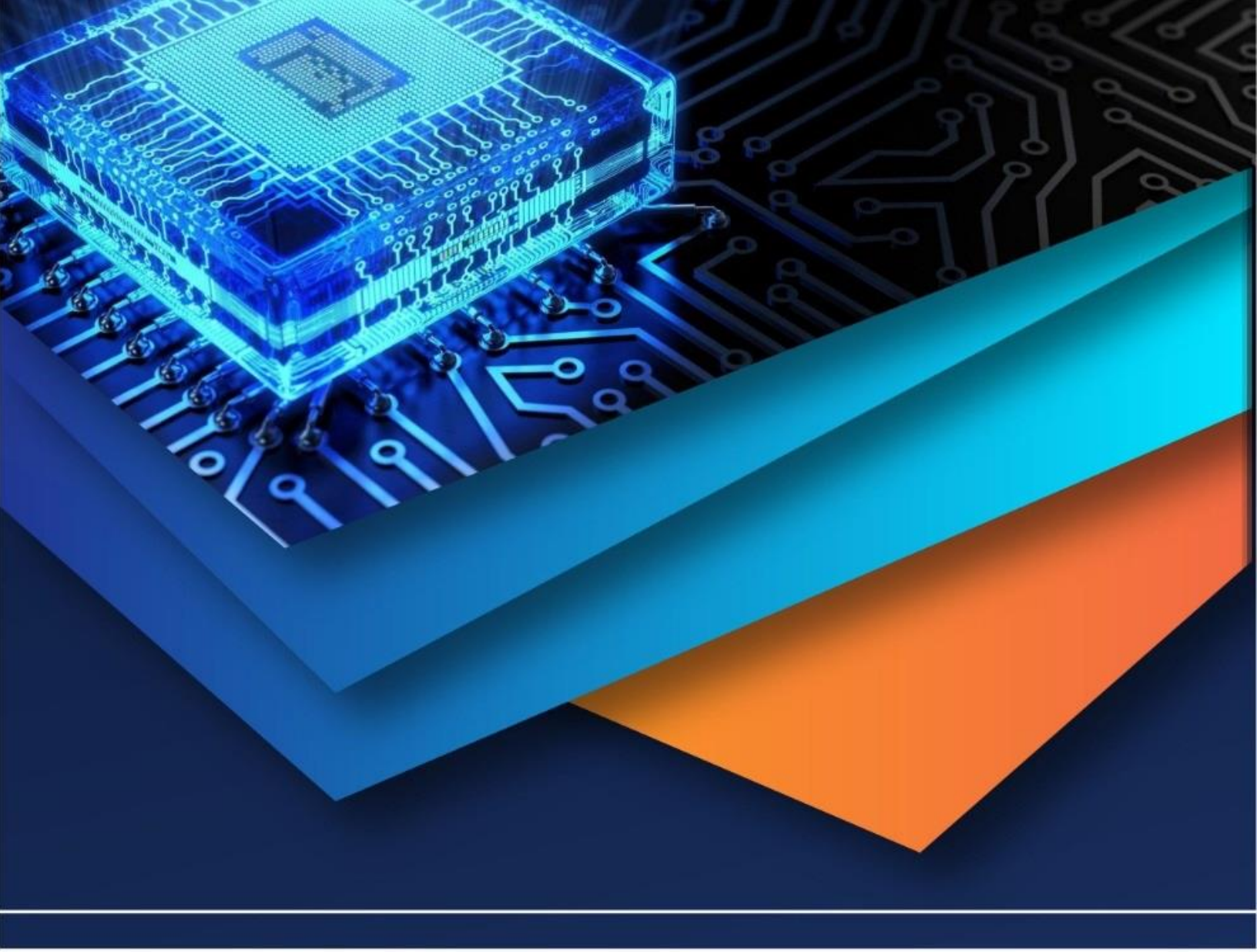
Looking ahead, there are several exciting possibilities for this project: (i) Language Expansion: Incorporating regional Indian languages like Telugu, Hindi, and Tamil through multi-language Tesseract models and gTTS language settings. (ii) Enhanced OCR Accuracy: Utilizing advanced deep learning-based OCR models such as EasyOCR and PaddleOCR to boost recognition capabilities in difficult lighting and with handwritten text. (iii) Voice Customization: Offering natural, adaptive speech output with neural TTS engines like Coqui TTS for a more human-like voice experience. (iv) Additional Assistive Features: Implementing navigation aids using GPS, object recognition with a camera and YOLO, and context awareness through scene description AI. (v) Large-Scale Deployment: Establishing accessible reading stations in schools, public libraries, government offices, and healthcare centers for permanent use.

X. ACKNOWLEDGMENT

The authors would like to express their heartfelt gratitude to Mr. S. Ramesh Babu, M.Tech., (Ph.D.), Assistant Professor in the Department of ECE at G. Pulla Reddy Engineering College (Autonomous) in Kurnool, for his invaluable guidance and unwavering support throughout this project. They also extend their thanks to Dr. K. Suresh Reddy, Ph.D., Professor and Head of the ECE Department at GPREC, Kurnool, for providing essential laboratory facilities and institutional backing. This project was undertaken as a Major Project requirement for the B.Tech. (ECE) program under the affiliation of JNTU Ananthapuramu.

REFERENCES

[1] World Health Organization (WHO), World Report on Vision, WHO Press, Geneva, Switzerland, 2019. [Online]. Available: https://www.who.int/publications/i/item/9789241516570
[2] OrCam Technologies Ltd., "OrCam MyEye 2 — Assistive Device for the Visually Impaired," Jerusalem, Israel, 2023. [Online]. Available: https://www.orcam.com
[3] eSight Corporation, "eSight 4 Electronic Glasses for the Visually Impaired," Ottawa, Canada, 2023. [Online]. Available: https://esighteyewear.com
[4] Raspberry Pi Foundation, Raspberry Pi 3 Model B+ Official Documentation, Cambridge, UK, 2023. [Online]. Available: https://www.raspberrypi.com/documentation/
[5] Google LLC, "gTTS — Google Text-to-Speech Python Library," Version 2.x, 2023. [Online]. Available: https://pypi.org/project/gTTS/
[6] UNESCO, Assistive Technology for Inclusive Education: Principles, Policies and Practices, UNESCO, Paris, France, 2020.
[7] A. Praveen Kumar, R. Suresh, and M. Lakshmi, "Assistive technology for visually impaired using OCR and speech synthesis," in Proc. IEEE Int. Conf. Electron., Comput. Commun. Technol. (CONECCT), Bangalore, India, 2022, pp. 1–6.
[8] V. Ramesh and K. Sundaravadivelu, "Smart reading device for visually impaired using Raspberry Pi and cloud OCR," Int. J. Eng. Res. Technol. (IJERT), vol. 10, no. 3, pp. 245–250, 2021.
[9] P. Jayaraman, G. Suresh, and T. Kumar, "Low-cost reading aid for blind people using OCR and TTS on embedded Linux platforms," in Proc. Int. Conf. Comput. Commun. Informatics (ICCCI), Coimbatore, India, 2020, pp. 1–5. Springer, Singapore.
[10] UNESCO, "Assistive Technology for Inclusive Education," Education Sector, UNESCO, Paris, 2020. [Online]. Available: https://unesdoc.unesco.org
[11] N. Katzir, "OrCam MyEye: A comprehensive assistive device for the visually impaired," IEEE Pervasive Comput., vol. 18, no. 1, pp. 68–71, Jan.–Mar. 2019.
[12] Microsoft Corporation, "Seeing AI — Talking Camera App for the Blind Community," Microsoft, Redmond, WA, 2023. [Online]. Available: https://www.microsoft.com/en-us/ai/seeing-ai
[13] NV Access, "NVDA (NonVisual Desktop Access) Screen Reader — Open Source," Version 2023.x, 2023. [Online]. Available: https://www.nvaccess.org
[14] Envision Technologies B.V., "Envision AI — Reading and Navigation for the Blind," Delft, Netherlands, 2023. [Online]. Available: https://www.letsenvision.com
[15] R. Smith, "An overview of the Tesseract OCR engine," in Proc. 9th Int. Conf. Document Anal. Recognition (ICDAR), Curitiba, Brazil, 2007, pp. 629–633. Google Inc. Tesseract OCR GitHub: https://github.com/tesseract-ocr/tesseract



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY