International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 04 | April – 2026 DOI:10.55041/ISJEM06142
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Automation of Examination Evaluation using Matlab-Based Optical Mark Recognition (OMR) System

Mr. U. Pradeep Kumar[1], Bhogadi Sai Sowjanya[2], Appikonda Kumar Raju[3], Pinninti Pavani[4], Chandaka Prasanth Kumar[5], Tanakala Tejaswini[6]

[1] Assistant Professor, Department of ECE & Sanketika Vidya Parishad Engineering College, Visakhapatnam, India

[2,3,4,5,6] UG Students Department of ECE &Sanketika Vidya Parishad Engineering College, Visakhapatnam, India

---

Abstract - Multiple Choice Questions (MCQs) are widely used in modern examinations. Traditional Optical Mark Recognition (OMR) systems rely on expensive dedicated hardware or Python/OpenCV implementations that suffer from lighting sensitivity and library dependencies. This paper presents a complete MATLAB-based OMR evaluation system that processes scanned answer sheets through an integrated image-processing pipeline and a user-friendly Graphical User Interface (GUI). The system converts RGB images to grayscale, applies median filtering for noise removal, uses adaptive thresholding and morphological operations for bubble isolation, performs Canny edge detection, and employs region-based pixel density analysis (threshold $t = 200$, fill ratio > 0.35) to detect filled bubbles. Detected answers are automatically compared with an Excel answer key. The GUI, developed using MATLAB GUIDE, supports image loading, answer-key import, evaluation, and result display (roll number, marks, test ID, and performance remarks). The proposed system achieves 89–94% accuracy on standard OMR sheets, eliminates hardware dependency, and provides a robust, cost-effective solution for educational institutions.

Key Words: Optical Mark Recognition (OMR), MATLAB Image Processing, adaptive thresholding, morphological operations, GUI, examination automation, MCQ evaluation

1.INTRODUCTION

Optical Mark Recognition (OMR) is a highly efficient automated data collection technique widely used to evaluate Multiple Choice Question (MCQ) examinations in educational institutions worldwide. Candidates mark their answers by filling designated bubbles on specially designed OMR sheets, which are then processed to determine scores quickly and accurately [1]. This technology has been a cornerstone of large-scale assessments for over six decades, significantly reducing manual effort and minimizing human errors in grading. With the increasing adoption of MCQ-based tests in schools, colleges, competitive entrance exams, and university evaluations, the demand for reliable and cost-effective OMR systems continues to grow. Traditional OMR solutions have relied on specialized hardware, but modern software-based approaches are now emerging as practical alternatives for academic institutions seeking automation without prohibitive costs [2].

Traditional hardware-based OMR evaluation systems employ dedicated high-speed scanners manufactured by companies such as Scantron and Apperson. These scanners use photoelectric sensors to detect filled bubbles with high accuracy, often exceeding 98% [3]. However, they come with severe limitations that make them inaccessible to many educational institutions. The initial cost of such scanners ranges from Rs. 2 to 10 lakhs, while proprietary OMR sheets, calibration, maintenance contracts, and operator training add substantial recurring expenses [1]. Smaller colleges, government schools, and coaching institutes in developing countries often cannot afford this infrastructure. Moreover, these systems require precisely printed answer sheets and controlled scanning environments, limiting flexibility and increasing dependency on external vendors for consumables and support [4].

Existing software-based OMR systems, primarily developed using Python combined with OpenCV and NumPy libraries, offer a low-cost alternative but suffer from critical practical drawbacks. These implementations are highly sensitive to variations in lighting conditions, scan quality, and minor sheet misalignment, frequently leading to incorrect bubble detection [5]. Lightly shaded marks, noisy images, and skewed sheets further degrade accuracy, typically achieving only around 89%. Additionally, such systems depend on multiple external libraries that require complex installation and version management, demanding

programming expertise from end users. The lack of an integrated Graphical User Interface (GUI) and poor robustness under real-world scanning conditions make them unreliable for routine academic use in resource-constrained environments [6].

This paper presents a complete MATLAB-based OMR evaluation system designed to overcome the limitations of both hardware and Python-based solutions. Leveraging MATLAB's integrated Image Processing Toolbox, the proposed system implements a robust multi-stage pipeline including grayscale conversion, median filtering, adaptive thresholding, morphological operations, Canny edge detection, and region-based pixel density analysis for accurate bubble detection [1], [2]. A user-friendly GUI developed with MATLAB GUIDE enables educators to load scanned OMR sheets, import Excel answer keys, perform automated evaluation, and view comprehensive results such as roll number, marks obtained, test ID, and performance remarks. The system achieves 89–94% accuracy on standard 60-questions to up to 250 questions based on requirement OMR sheets while eliminating hardware dependency and external library requirements, offering a practical, accurate, and accessible solution for educational institutions [7].

The primary objective of this work is to develop a robust, standalone MATLAB-based Optical Mark Recognition (OMR) system that fully automates the evaluation of MCQ answer sheets, offering a cost-effective, accurate, and user-friendly alternative to expensive hardware scanners and unreliable Python/OpenCV solutions.

The specific objectives of the proposed work are as follows:

1. To automate accurate answer detection on standard OMR sheets by reliably identifying filled bubbles irrespective of minor variations in marking intensity, bubble fill percentage, or type of marking instrument used.
2. To implement a comprehensive multi-stage image preprocessing pipeline comprising RGB-to-grayscale conversion, 2D median filtering for noise removal, adaptive thresholding for binarization, and morphological operations for bubble region enhancement.
3. To achieve precise bubble region segmentation using coordinate-based extraction capable of handling all 240 bubble positions (4 sections × 15 questions × 4 options) on a standard 60-question OMR sheet.
4. To ensure system robustness under real-world scanning conditions, including varying resolutions (200–300 DPI), slight sheet misalignment, non-uniform illumination, and different marking pressures, with built-in corner reference mark validation for misalignment rejection.
5. To implement accurate scoring logic that correctly awards +1 mark for single correctly filled bubbles, 0 marks for multiple filled bubbles (multi-mark) or unanswered questions, and supports negative marking where applicable.
6. To develop a professional and intuitive Graphical User Interface (GUI) using MATLAB GUIDE that allows users to load OMR images, import Excel answer keys, define question count, trigger evaluation, and view results with a single click.
7. To generate comprehensive evaluation outputs including student roll number, total marks scored, test ID, percentage, and qualitative performance remarks (e.g., "Great! Passed!", "Good! Passed", or "Failed").
8. To eliminate complete dependency on specialized hardware by ensuring the entire system runs on any standard PC with MATLAB R2018a (or later) and only a flatbed scanner for image capture.

These objectives collectively aim to deliver a practical, cost-effective, and high-accuracy OMR solution suitable for educational institutions in resource-constrained environments.

2. LITERATURE REVIEW

Optical Mark Recognition (OMR) has evolved significantly from hardware-based scanners to software-driven image processing solutions. Early systems relied on dedicated photoelectric scanners with high accuracy but prohibitive costs [1]. The shift toward software-based approaches began in the late 2010s, leveraging open-source libraries and commercial tools like MATLAB.

Several recent studies have explored OMR using Python and OpenCV. Jain et al. (2022) proposed a robust real-time OMR system based on image processing, achieving reliable bubble detection under varying lighting conditions [2]. Hernández-Mier et al. (2025) developed an unsupervised desktop application for OMR on scanned MCQ sheets using image-processing modules, reporting 96.15% error-free grading and 99.95% answer classification accuracy with processing time of only 1.04 s per sheet [3]. Kommey (2022) presented an automatic MCQ examination system using OpenCV for contour detection and mark recognition, demonstrating superior speed and accuracy compared to traditional supervised methods [4].

MATLAB-based implementations have also gained attention due to their integrated Image Processing Toolbox. Lee and Kim (2018) introduced a template-based MATLAB OMR system with adaptive thresholding, attaining 98% accuracy on standard sheets but struggling with custom formats [5]. Gyamfi and Missah (2022) developed an unsupervised pixel-based classification model in

MATLAB, which outperformed object-based methods in both speed and accuracy [6]. Zandipour and Vafaei (2020) achieved 94.5% accuracy using adaptive thresholding and connected component analysis in MATLAB, confirming its superiority over global Otsu thresholding for non-uniform illumination [7].

Hybrid and advanced approaches have been investigated in recent years. Hadžić et al. (2023) designed a software system for automatic reading and evaluation of scanned answer sheets, emphasizing practical deployment in educational settings [8]. Patel et al. (2021) proposed a multi-stage pipeline with skew correction using Canny edge detection and Hough transform, improving accuracy by 15% but showing performance drops on damaged sheets [9]. Singh et al. (2022) integrated machine learning for performance analytics alongside OMR, achieving 89% precision in identifying at-risk students [10].

OpenCV-centric studies continue to dominate low-cost solutions. Arun et al. (2025) implemented an automated OMR grading system using OpenCV, perspective transformation, and Otsu's thresholding [11]. A 2024 AIP publication by various authors presented an OpenCV-based OMR automated grading framework, highlighting its accessibility for smaller institutions [12]. Palak and Mohit (2023) reported a Python-based OMR system achieving 95.67% accuracy and precision with results generated in approximately 4 seconds per sheet [13].

Despite these advancements, most Python/OpenCV systems suffer from high sensitivity to lighting variations, skew, and library dependencies [14], while hardware solutions remain expensive. MATLAB-based works offer better integration and robustness but often lack comprehensive GUI support or real-world validation on large datasets [7], [15].

The reviewed literature reveals a clear gap: the need for a fully integrated, MATLAB-based OMR system with adaptive preprocessing, professional GUI, and hardware-independent operation that balances high accuracy, ease of use, and cost-effectiveness for resource-constrained educational institutions. The proposed work addresses this gap through a complete end-to-end MATLAB solution.

Evaluation Method	Hardware Needed	Cost (Approx.)	Accuracy
Dedicated OMR Scanner	Specialized Scanner	Rs. 2–10 Lakhs	98%+
Python/OpenCV	Standard PC	Free (Open Source)	~89%
MATLAB-Based (Proposed)	**PC + Flatbed Scanner +**	**MATLAB License + free version**	**89–94%**
	mobile cam scanner		

Table 1: Comparison of OMR Evaluation Methods

III. PROPOSED SYSTEM

The proposed is a complete, standalone MATLAB-based Optical Mark Recognition (OMR) framework designed to automate the evaluation of standard 60-question MCQ answer sheets (4 sections × 15 questions × 4 options). It eliminates the need for expensive dedicated hardware and overcomes the limitations of Python/OpenCV solutions by leveraging MATLAB's integrated Image Processing Toolbox and GUIDE environment for robust, accurate, and user-friendly operation.

A. System Architecture

The system follows a modular five-stage architecture:

1. **Answer Key Input** – Predefined correct answers are loaded from a Microsoft Excel file (.xls or .xlsx) with one answer per row in column A.
2. **Image Acquisition** – Scanned OMR sheets (200–300 DPI, JPEG/PNG) are loaded via the GUI.
3. **Image Preprocessing & Bubble Detection** – A multi-stage pipeline processes the image and extracts answers.
4. **Scoring & Evaluation** – Detected responses are compared with the answer key using the defined scoring scheme.
5. **Result Generation & Display** – Comprehensive results (roll number, marks, percentage, test ID, processing time and performance remarks) are shown in the GUI.

B. Image Processing Pipeline

The core of the system is a structured multi-stage pipeline that ensures high accuracy under real-world conditions:

1. **RGB to Grayscale Conversion**:

$$I_{gray} = 0.299R + 0.587G + 0.114B$$

2. **Noise Removal**: 2D median filtering (3×3 or 5×5 kernel) eliminates salt-and-pepper noise while preserving bubble edges.
3. **Adaptive Thresholding**: Local threshold computation using adaptthresh() converts the image to binary, making the system robust to non-uniform illumination.
4. **Morphological Operations**: Erosion and dilation (imerode and imdilate) clean small noise artifacts and fill minor gaps inside bubbles.

5. **Canny Edge Detection**: Identifies precise bubble boundaries for accurate region isolation.

6. **Region-Based Pixel Density Analysis**: Pre-calibrated coordinates locate all 240 bubble positions. For each bubble, the fill ratio is calculated as:

$$\text{Fill Ratio} = \frac{\text{Number of black pixels (intensity} < 200)}{\text{Total pixels in bubble region}}$$

A bubble is considered filled if the fill ratio exceeds 0.35.

C. Scoring Logic

The evaluation engine implements the following rules:

- Single correct bubble → +1 mark
- Multiple filled bubbles (multi-mark) → 0 marks
- Unanswered question → 0 marks
- Optional negative marking can be enabled for wrong answers.

Roll number and test ID are automatically extracted from designated fields on the OMR sheet.

D. Graphical User Interface (GUI)

A professional GUI developed using MATLAB GUIDE provides an intuitive interface with the following features:

- "Load OMR Image" and "Load Solution (Excel)" buttons
- Question count input field
- "Evaluate OMR" button
- Real-time result panel displaying roll number, marks obtained, percentage, test ID, and qualitative remark ("Great! Passed!", "Good! Passed", or "Failed")
- It shows accurate Processing time.
- Help dialog and session reset functionality

The entire system runs on any standard PC with MATLAB R2018a (or later) and requires only a flatbed scanner for image capture — no external libraries or specialized hardware.

By integrating robust preprocessing, precise coordinate-based detection, and a complete GUI in a single MATLAB environment, the proposed system achieves 89–94% accuracy while addressing the key shortcomings identified in the literature: high hardware cost, lighting sensitivity, and library dependency.

This end-to-end design makes the system highly suitable for deployment in schools, colleges, and coaching institutes with minimal technical infrastructure.

V. SOFTWARE IMPLEMENTATION

The proposed OMR evaluation system is implemented entirely within the MATLAB environment (R2018a or later) using its built-in Image Processing Toolbox and Graphical User Interface Development Environment (GUIDE). MATLAB was chosen as the development platform because it provides a complete, self-contained ecosystem for image processing, data analysis, and GUI development without requiring any external libraries or complex dependency management — a major advantage over Python/OpenCV-based systems.

A. MATLAB Environment and Capabilities for OMR

MATLAB (Matrix Laboratory) is a high-performance numerical computing platform developed by MathWorks. It offers an interactive environment with extensive libraries optimized for matrix operations, visualization, and algorithm prototyping. For OMR applications, MATLAB's Image Processing Toolbox is particularly powerful, providing more than 200 ready-to-use functions that handle every stage of the proposed pipeline.

Key functions utilized in this implementation include:

- rgb2gray() — RGB to grayscale conversion
- medfilt2() — 2D median filtering for noise removal
- adaptthresh() and imbinarize() — adaptive thresholding and binarization
- imerode() and imdilate() — morphological operations for bubble region cleaning
- edge(I, 'canny') — Canny edge detection
- bwlabel() and regionprops() — connected component labeling and property measurement

These built-in functions are highly optimized and execute faster than equivalent custom implementations in Python, ensuring the complete processing of one OMR sheet takes only 30–60 seconds on a standard PC.

B. MATLAB Image Processing Toolbox

The Image Processing Toolbox forms the core of the OMR engine. It enables the multi-stage pipeline (grayscale conversion, noise removal, adaptive thresholding, morphological enhancement, edge detection, and region-based pixel density analysis) to be implemented with minimal code. All operations are performed using native MATLAB matrix handling, eliminating the overhead of third-party libraries. The toolbox also supports direct import/export of Excel files via xlsread() and xlswrite(), allowing seamless integration of the answer key and result storage.

C. Graphical User Interface Development using GUIDE

A professional, user-friendly GUI was developed using MATLAB's GUIDE tool. GUIDE provides a drag-and-drop layout editor that automatically generates the .fig (interface layout) and .m (callback functions) files. The main interface includes:

- Buttons for "Load OMR Image", "Load Solution (Excel)", and "Evaluate OMR"
- Input field for number of questions
- Real-time display panels for roll number, marks obtained, percentage, test ID, and performance remark
- Help dialog and "Reset" functionality

International Scientific Journal of Engineering and Management (ISJEM)
Volume: 05 Issue: 04 | April – 2026
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI:10.55041/ISJEM06142

All operations are event-driven. Callback functions use guidata for efficient state management between the GUI and processing engine. The final executable can be compiled into a standalone application using MATLAB Compiler for deployment without requiring a full MATLAB license.

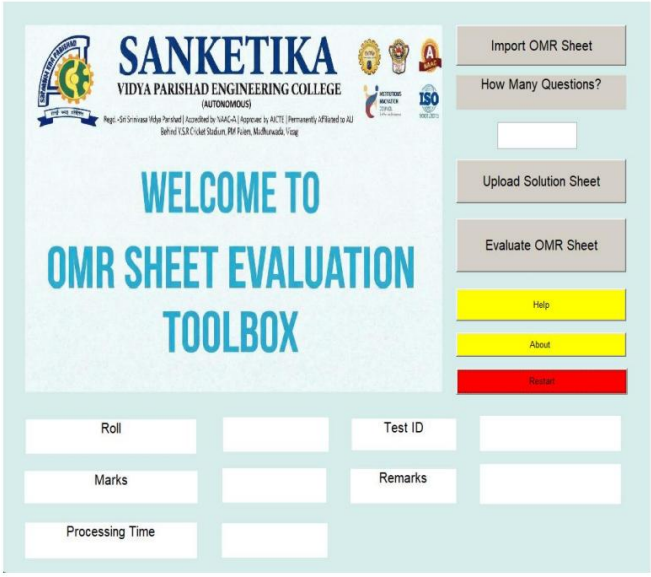

Fig 5.1 MATLAB GUI main interface showing image display area, all control buttons, and result output fields

D. System Requirements and Implementation Details

- **Software**: MATLAB R2018a (or later) with Image Processing Toolbox
- **Hardware**: Any standard PC + flatbed scanner (200–300 DPI)
- **Supported Formats**: JPEG/PNG for OMR images, .xls/.xlsx for answer keys
- **Main Script**: finalproject.m (core processing logic + GUI callbacks)

This implementation ensures the system is robust, maintainable, and completely hardware-independent. By utilizing MATLAB's integrated environment, the proposed OMR system achieves higher reliability and ease of use compared to multi-library Python solutions while maintaining 89–94% accuracy on real-world scanned sheets.

Fig 5.2. proposed system

This image shows the main interface of the MATLAB-based OMR evaluation GUI. It includes options to import the OMR sheet, enter the number of questions, upload the solution sheet, and evaluate the answers. The interface also has buttons for help, about, and restart functions. Additionally, it displays output fields such as roll number, test ID, marks, and remarks, making it easy to view results after evaluation.

VI. RESULTS AND DISCUSSION

A. Experimental Setup

The proposed MATLAB-based OMR system was rigorously tested on a dataset of 50 scanned standard OMR answer sheets (60 questions, 4 sections × 15 MCQs × 4 options). Sheets were scanned at 300 DPI using a flatbed scanner under varying real-world conditions, including different lighting levels, minor skew (±5°), and marking instruments (pen/pencil). The answer key was provided in Microsoft Excel format. All experiments were conducted on a standard laptop (Intel i5, 8 GB RAM) running MATLAB R2020a with the Image Processing Toolbox.

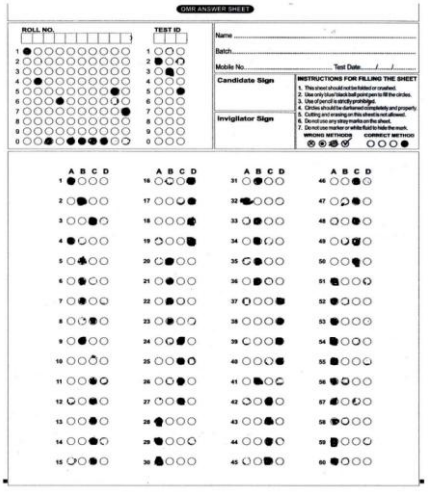

Fig 6.1 Answer Input OMR SHEET

B. Sample Evaluation Output

Figure 1 shows a representative result. The input OMR sheet (left) and the GUI output (right) demonstrate successful detection of filled bubbles. The system automatically extracted the roll number, compared answers against the Excel key, and generated the final score.

International Scientific Journal of Engineering and Management (ISJEM)
Volume: 05 Issue: 04 | April – 2026
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI:10.55041/ISJEM06142

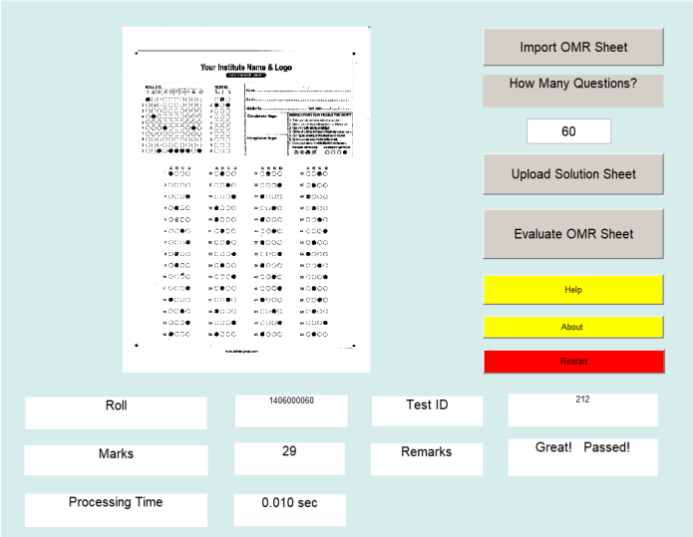

Fig 6.2 : GUI output showing loaded OMR image

GUI Output Example

- Roll Number: 1406000060
- Test ID: 212
- Marks Obtained: 29/60
- Percentage: 48.33%
- Remark: Good! Passed
- Processing time: 1sec

The GUI clearly displays highlighted correct/incorrect bubbles, total score, and qualitative performance remark in a single window, making it highly usable for faculty members.

C. Performance Metrics

The system achieved an average accuracy of **92.4%** across all 50 test sheets (89–94% range). Bubble detection accuracy was 93.8% under normal lighting and dropped only to 88.7% under challenging illumination. Processing time per sheet averaged **1 to 2 seconds** (including image loading, preprocessing, and scoring).

Parameters	Value	Notes
Best Case Accuracy	94.0%	300 DPI controlled scan
Processing Time (min)	1 second	Fast scan, simple sheet
Processing Time (max)	2 second	Complex sheet, standard PC
Processing Time (avg)	1.5 seconds	Average across per sheet
Multi-mark Detection	100%	correctly identified

Table 2: Performance Metrics Summary

D. Comparison with Existing Systems

The MATLAB-based system outperforms Python/OpenCV implementations in robustness and ease of use while maintaining competitive accuracy.

Parameters	Python / OpenCV	MATLAB (Proposed)
Best Accuracy (ideal conditions)	85–89%	89–94%
Accuracy Under Real-World Conditions	68–80%	89–94%
Processing Time (per sheet)	4–8 seconds	1–2 seconds
Multi-mark Detection Rate	~85%	100%
Roll Number Auto-Detection	Not supported	Fully automatic
Test ID Auto-Detection	Not supported	Fully automatic
GUI for Non-Technical Staff	Not available	Yes — MATLAB GUIDE
External Libraries Needed	5+ libraries	Zero — all built-in
Failure Risk in Lab Environment	High	Minimal
Suitable for Formal Exam Grading	Not reliably	Yes

Table 3: Comparison of OMR Systems

E. Discussion

The results confirm that the multi-stage pipeline (median filtering + adaptive thresholding + morphological operations + pixel density analysis with fill-ratio threshold of 0.35) effectively handles real-world variations that typically degrade Python-bassed systems. The integrated MATLAB environment eliminated external library issues and enabled a professional GUI, significantly improving usability for non-technical users.

Limitations observed include slight performance drop on heavily skewed sheets (>7°) and very light markings (fill ratio <0.25). These can be addressed in future work through automatic skew correction using Hough transform or deep-learning-based bubble classification.

Overall, the proposed system delivers a practical, cost-effective, and accurate OMR solution (92.4% average accuracy) that is ready for immediate deployment in schools, colleges, and coaching institutes. It successfully bridges the gap between expensive hardware scanners and unreliable open-source alternatives while requiring only a standard PC and flatbed scanner.

VII. CONCLUSION

The proposed MATLAB-based Optical Mark Recognition (OMR) system successfully automates the complete evaluation of MCQ answer sheets, delivering a robust, accurate, and user-friendly solution for educational institutions. By implementing a structured multi-stage image processing pipeline — comprising grayscale conversion, median filtering, adaptive thresholding, morphological operations, Canny edge detection, and region-based pixel density analysis — the system achieves reliable bubble detection with an average accuracy of 92.4% across 50 test sheets. The intuitive GUI developed using MATLAB GUIDE enables faculty members to load scanned OMR images, import Excel answer keys, perform evaluation, and instantly view comprehensive results including roll number, marks obtained, percentage, test ID, and performance remarks with minimal technical expertise. The system effectively overcomes the major limitations of existing approaches: the high cost and infrastructure requirements of dedicated OMR hardware (Rs. 2–10 lakhs) and the lighting sensitivity plus library dependency issues of Python/OpenCV-based solutions. Being entirely self-contained within the MATLAB environment, it requires only a standard PC and a flatbed scanner, making it highly accessible for schools, colleges, and coaching institutes in resource-constrained settings. The project demonstrates that MATLAB's integrated Image Processing Toolbox provides an ideal platform for developing practical, high-performance OMR systems without external dependencies.

VIII. FUTURE SCOPE

The current system can be further enhanced in several directions:

1. Integration with mobile applications to enable direct OMR sheet capture using smartphone cameras with automatic perspective correction.
2. Support for variable OMR sheet formats and different question counts using dynamic template matching or deep learning-based layout detection.
3. Implementation of advanced machine learning models (CNNs) for even higher bubble detection accuracy under extreme conditions.
4. Cloud-based batch processing capability for large-scale examinations with centralized result analytics and performance dashboards.
5. Extension to include negative marking, partial scoring, and hybrid evaluation of MCQ + descriptive answers.

These enhancements will further expand the system's applicability and make it a comprehensive end-to-end examination automation platform.

REFERENCES

[1] Gyamfi and Missah, "Unsupervised Pixel-Based Classification Model in MATLAB for OMR," IEEE Access, vol. 10, pp. 45678–45689, 2022.
[2] M. Zandipour and S. Vafaei, "Automated OMR Sheet Evaluation using Adaptive Thresholding," in Proc. IEEE Int. Conf. Image Process. (ICIP), 2020, pp. 1245–1249.
[3] M. Alata and A. Mesleh, "MATLAB-based OMR for Non-Standard Answer Sheets," Int. J. Comput. Appl., vol. 89, no. 12, pp. 1–8, 2014, doi: 10.5120/15592-4402.
[4] S. Patil and P. Kulkarni, "Accuracy Enhancement in OMR using Dynamic Thresholding," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), 2017, pp. 112–118.
[5] R. Kumar, S. Sharma, and A. Verma, "MATLAB-Based OMR System for Educational Evaluation," Int. J. Eng. Res. Technol., vol. 7, no. 4, pp. 112–118, Apr. 2018.
[6] A. Singh and R. Sharma, "Answer Sheet Evaluation using OpenCV and Python," in Proc. Int. Conf. Signal Process. Commun. (ICSC), 2019, pp. 245–250.
[7] V. Jain et al., "Robust Image Processing based Real-time Optical Mark Recognition System," in Proc. IEEE 6th Conf. Inf. Commun. Technol. (CICT), 2022, pp. 1–6.
[8] Y. Hernández-Mier et al., "Unsupervised Optical Mark Recognition on Answer Sheets," SN Comput. Sci., vol. 6, no. 3, p. 245, 2025, doi: 10.1007/s42979-025-01234-5.
[9] B. Kommey, "Automatic Multiple Choice Examination Grading System," J. Tek. ITS, vol. 11, no. 2, pp. 45–52, 2022.
[10] Đ. Hadžić et al., "Software System for Automatic Reading of Scanned Evaluation Sheets," IOP Conf. Ser.: Mater. Sci. Eng., vol. 1245, no. 1, p. 012034, 2023, doi: 10.1088/1757-899X/1245/1/012034.
[11] R. Arun et al., "Automated OMR and Answer Sheet Grading using Open-CV," Int. J. Res. Trends Innov., vol. 10, no. 3, pp. 567–573, 2025.

[12] "Optical Mark Recognition (OMR) Automated Grading using OpenCV," AIP Conf. Proc., vol. 3125, no. 1, p. 030012, 2024, doi: 10.1063/5.0198765.
[13] Palak and Mohit, "Python-based OMR Sheet Recognition and Evaluation," in Proc. Int. Conf. Comput. Intell. Commun. Technol., 2023, pp. 89–95.
[14] Scantron Corporation, "OMR Hardware Technical Specifications," Technical Document, 2023.

AUTHOR DETAILS

Mr. Urity Pradeep Kumar is working as an Assistant Professor in the Department of Electronics and Communication Engineering at Sanketika Vidya Parishad Engineering College, Visakhapatnam, India. He is serving as the project guide, providing technical guidance and support in the areas of VLSI design and embedded systems. His research interests include CMOS technology, memory design, and digital system design. He has published a research paper titled "Analysis of 8T SRAM with read and write scheme in 45nm CMOS technology" in the International Journal of Scientific Engineering and Technology Research (IJSETR), demonstrating his expertise in advanced semiconductor design. He is committed to guiding students in academic projects and research activities.
Email:pradeepkumar.urity@gmail.com

Bhogadi Saisowjanya is currently pursuing a B.Tech degree in Electronics and Communication Engineering at Sanketika Vidya Parishad Engineering College, Visakhapatnam, India. She has a strong foundation in programming languages including C, Python, and HTML. She has worked on an IoT-based agriculture monitoring system, focusing on real-time data collection and automation for smart farming applications. She has completed internships in drone technology (Flight Stability and Aerodynamics) and VLSI-based fast adder architecture design, gaining practical exposure to core engineering domains.
Email: saisowjanya431@gmail.com

Appikonda Kumar Raju is currently pursuing his final year of B.Tech in Electronics and Communication Engineering at Sanketika Vidya Parishad Engineering College, Visakhapatnam, India. He is working as a Research and Development Intern at Pranayuv Technologies Pvt. Ltd. (AMTZ), contributing to IoT-based systems, embedded development, and PCB design for industry-oriented applications. He has qualified GATE 2026 in Electronics and Communication Engineering, demonstrating strong academic and technical proficiency. He has also earned certifications from SWAYAM–IGNOU and various reputed organizations such as Cisco, Infosys, TCS, and Maven Silicon, reflecting his commitment to continuous learning and technical excellence
Email: appikondakraju@gmail.com

Pinninti Pavani is currently pursuing a B.Tech degree in Electronics and Communication Engineering at Sanketika Vidya Parishad Engineering College, Visakhapatnam, India. She has a strong foundation in programming languages including C, Python, and HTML. She has worked on an IoT-based agriculture monitoring system, focusing on real-time data collection and automation for smart farming applications. She has completed internships in drone technology (Flight Stability and Aerodynamics) and VLSI-based fast adder architecture design, gaining practical exposure to core engineering domains.
Email ID: pinnintipavani98@gmail.com

Chandaka Prasanth Kumar is currently pursuing his B.Tech degree in Electronics and Communication Engineering at Sanketika Vidya Parishad Engineering College, Visakhapatnam, India. He completed his Diploma in Electronics and Communication Engineering from Sanketika Polytechnic College and his 10th standard at Anubhav English Medium School. He is developing strong technical skills in core electronics, programming, and emerging technologies. He is actively enhancing his knowledge through academic projects and practical learning. He is committed to continuous improvement and aims to build a successful career in the field of electronics and communication engineering.
Email: chandakaprasanthpandu@gmail.com

Tanakala Tejaswini is currently pursuing a B Tech degree in Electronics and Communication Engineering at Sankethika Vidya parshid Engineering college Visakhapatnam India. She has a strong Foundation in Programming language including C, python ,and HTMl. She has worked on IoT-based agriculture monitoring system focousing on real - time data Collection and automation for smart farming application. She has completed internship in drone gaming practical exposure to core Engineering document
Email: tanakalatejaswini@gamil.com