University of Windsor
Scholarship at UWindsor

Electronic Theses and Dissertations Theses, Dissertations, and Major Papers

2023

Webcam-Based Optical Mark Reader for Grading Assessments

Muhammad Fahad Zafar
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Part of the Computer Sciences Commons

Recommended Citation

Zafar, Muhammad Fahad, "Webcam-Based Optical Mark Reader for Grading Assessments" (2023). *Electronic Theses and Dissertations*. 8963.
https://scholar.uwindsor.ca/etd/8963

This online database contains the full-text of PhD dissertations and Masters’ theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.

Webcam-based Optical Mark Reader for Grading Assessments

By

Muhammad Fahad Zafar

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2023

© 2023 Muhammad Fahad Zafar

Webcam-based Optical Mark Reader for Grading Assessments

By

Muhammad Fahad Zafar

APPROVED BY:

F. Baki
Odette School of Business

I. Ahmad
School of Computer Science

J. Chen, Advisor
School of Computer Science

January 18, 2023

Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

Optical Mark Recognition (OMR) is a process of retrieving information from the forms where users put marks in particular locations. OMR technique has been improved significantly since its inception, moving from specialized hardware-based solutions to more adaptable and sophisticated software-based solutions using a scanner. Recently, webcam-based software solutions have emerged because of their affordability and ease of use. In this thesis, we propose an improved webcam-based OMR solution for grading the student tests given on a specifically formatted answer sheet. We apply various image processing techniques to achieve effective grading, resolving technical problems stemming from multiple factors like image orientation, skewness, and uneven illumination. In particular, we propose a new variant of the Otsu binarization technique for image segmentation to effectively separate the background and foreground. The proposed grading solution aims at maximizing the criteria for acceptable operations so that 100% accurate grading among those accepted ones can be achieved. The evaluation of the proposed method is two-fold: (i) the proposed variant of Otsu is compared with other binarization algorithms for its effectiveness in background separation; (ii) the grading solution is compared with other state-of-the-art webcam-based solutions where the individual contribution of each applied image processing technique to the grading enhancement will be identified.

Dedication

I dedicate this thesis to my parents, teachers, siblings, and friends for all their efforts, prayers, and support. Mainly my brother and my sister-in-law who supported me emotionally, cared for me unconditionally, stood by me through thick and thin, and shared their experiences and words of wisdom which really helped me.

Last but certainly not least, I want to thank and dedicate this thesis to my manager at work, Wesley Small. He allowed me to work flexibly and always cared and understood my situations and struggles as an International Student. Without him, I would have not been able to complete this work on time.

I am genuinely grateful for having all of you in my life.

Acknowledgements

I owe Dr. Chen, my advisor, a lot of appreciation for her unceasing support and insights. Her expertise led me in the right direction from day one, which made it possible for me to complete this thesis.

I am also grateful to Dr. Imran Ahmad, my internal reader, and Dr. Fazle Baki, my external reader, for their valuable time and comprehensive professional guidance, which made me improve my thesis.

Thank you all for your perseverance, inspiration, zeal, and vast knowledge.

Table of Contents

Declaration of Originality iii

Abstract iv

Dedication v

Acknowledgements vi

List of Tables x

List of Figures xi

Chapter 1 – Introduction 1

1.1 Optical Mark Recognition 1

1.2 Types of OMR 2

1.2.1 Hardware-based OMR 2

1.2.2 Software-based OMR 3

1.2.3 Webcam-based OMR 3

1.3 Technical Difficulties in Webcam-based OMR 4

1.4 Thesis Motivation 4

1.5 Problem Statement and Thesis Contributions 5

1.6 Thesis Outline 6

Chapter 2 – Background Study 7

2.1 Image Processing Terminology 7

2.2 Image Binarization Algorithms 9

2.2.1 Simple Image Binarization 9

2.2.2 Adaptive Image Binarization 9

2.2.3 Otsu Binarization 10

2.2.4 Niblack Binarization 11

2.2.5 Sauvola Binarization 11

2.2.6 Wolf & Jolion Binarization 11

2.3 OpenCV and its algorithms 12

2.3.1 SimpleBlobDetector 13

2.3.2 FindContours 14

2.3.3 MinAreaRect 15

Chapter 3 – Related Work 16

3.1 Paper 1 – Shape Matching OMR 16

3.2 Paper 2 – Template Matching OMR 18

3.3 Paper 3 – Mobile Camera OMR 20

3.4 Paper 4 – Machine Learning OMR 21

3.5 Paper 5 – Enhanced Algorithm for OMR 22

3.6 Paper 6 – Three-Scaled Images OMR 23

Chapter 4 – Methodology 25

4.1 Stage 1 – Pre-Processing 25

4.1.1 BGR to Grayscale 26

4.1.2 Grayscale Image Resizing 26

4.1.3 Contrast Enhancement 27

4.1.4 Image Smoothening 28

4.1.5 Image Binarization 29

4.2 Stage 2 – ROI Extraction 34

4.2.1 Finding Contours & making rectangles 34

4.2.2 Cropping Image with Largest Rectangle 36

4.3 Stage 3 – Blob Detection 37

4.3.1 Finding Blobs....37

4.3.2 Validating Blobs....38

4.3.3 Orientation Correction....39

4.3.4 Blob Points Rotation....40

4.3.5 Results Compilation....40

Chapter 5 – Experiments and Results....43

5.1 Environmental Setup....43

5.2 Dataset....44

5.3 Evaluation Criteria....44

5.3.1 Effectiveness of Proposed Binarization Algorithm....45

5.3.2 Effectiveness of Complete Proposed System....45

5.4 Results....46

5.4.1 Proposed Binarization Algorithm Test....46

5.4.2 Proposed System Complete Test....51

5.4.3 Comparison of the Proposed System with Existing Systems....59

5.5 Observations from Experiments....60

Chapter 6 – Conclusion and Future Work....61

Bibliography....62

Vita Auctoris....65

List of Tables

Table 1: Sample Format of Input File with Correct Answers.................................... 42

Table 2: Sample Format of Output File with Markings... 42

Table 3: Environmental Setup 43

Table 4: Dataset Composition.. 44

Table 5: Confusion Matrix for finding the accuracy of an Individual Sheet 45

Table 6: Pre-defined Value used for Binarization Algorithms 49

Table 7: Results to test basic functionality 51

Table 8: Results to test orientation 52

Table 9: Results to test skewness.. 53

Table 10: Results to test the combination of skewness and orientation 54

Table 11: Results to test lighting conditions... 55

Table 12: Results to test the combination of lighting conditions and orientation 56

Table 13: Results to test the combination of lighting conditions and skewness......... 57

Table 14: Results to test the complete dataset 58

Table 15: Results of other binarization algorithms.. 58

Table 16: Comparison between the proposed system and existing methods............. 59

List of Figures

Figure 1: Example of a Coloured Image's Pixels 7

Figure 2: Intensity Information of each Gray Level 7

Figure 3: Example of a Binary Image 8

Figure 4: Pixel Connectivity Neighbourhoods 8

Figure 5: Example of Simple and Adaptive Binarization 10

Figure 6: Otsu Binarization Algorithm 10

Figure 7: Example of Niblack, Sauvola, Wolf & Jolion Binarization Algorithms 12

Figure 8: Example of OpenCV's FindContours Approximation Methods 14

Figure 9: Example of OpenCV's MinAreaRect 15

Figure 10: Shape Matching OMR 16

Figure 11: Template Matching OMR 19

Figure 12: Flowchart of Mobile-based OMR 20

Figure 13: Algorithms used in Machine Learning OMR 21

Figure 14: Block Diagram of Three-Scaled Images OMR 23

Figure 15: Proposed Webcam-OMR System 25

Figure 16: Conversion of a BGR Image to a Grayscale Image 26

Figure 17: Example of Aspect Ratio Importance 27

Figure 18: Example of Histogram Equalization 27

Figure 19: Example of CLAHE Effect 28

Figure 20: Example of 5x5 Gaussian Filter Effect 29

Figure 21: Examples of Otsu Binarization 31

Figure 22: Examples of different Otsu variants 33

Figure 23: Total Mean computed on Dark and Bright Images 34

Figure 24: Example of Rectangular Contours 35

Figure 25: Example of a Rectangle Contour on Page Edges 35

Figure 26: Working of Affine Transformation 36

Figure 27: Example of a Cropped Image with Incorrect Orientation 37

Figure 28: Example of a blob-detected image 38

Figure 29: Example 1 of different binarization algorithms 47

Figure 30: Example 2 of different binarization algorithms 48

Figure 31: Chart showing a comparison between binarization algorithms................. 49

Figure 32: Chart showing the time taken by different binarization algorithms 50

Chapter 1 – Introduction

1.1 Optical Mark Recognition

Optical Mark Recognition (OMR) or Optical Mark Reader is an electronic process of extracting and interpreting information marked by humans in specific places onto a piece of paper. OMR is also called "mark sensing" and the document from which data is extracted is called an OMR or bubble sheet.

There are several applications of OMR technology. It is most commonly used for grading multiple choice assessments by educational institutes, according to Zhang et al. [1]. Also, it is used by data collection agencies for conducting surveys and collecting feedback forms, voting, timesheets, geocoding, tracking product inventories etc.

With such technological advancements in the modern world, one might ask about the need for OMR and paper forms in the first place. Especially in this post-pandemic era, people are trained enough to use online resources and do their jobs without needing paper. Still, roughly 37 percent of the world's population, primarily residing in developing countries, has never used the Internet, according to the data from Nov 2021 [2]. This staggering figure elaborates on the importance and application of paper-based forms and OMR.

Data processing centers have been using OMR technology since the 1960s for faster and more accessible information processing [3]. In the early days, we had only hardware-based solutions which needed a dedicated scanning device and special pre-printed sheets to function. However, since the 1990s, software-based solutions have taken off, replacing the need for expensive hardware-based solutions.

1.2 Types of OMR

1.2.1 Hardware-based OMR

In the early 1930s, a high school science teacher from Michigan named Reynold Johnson designed an electronic test-scoring machine for his personal use. His machine could read pencil marks from the specially prepared paper and compile those markings by comparing them to an answer key set up on the machine to calculate the final score. Later, in 1934, he was hired by IBM, where he transformed his device into a production model called IBM-805 Test Scoring Machine, which debuted in 1938 [4].

IBM-805 Test Scoring Machine could mark answer sheets ten times faster than manual markings and around 800 sheets per hour if operated by an experienced person. Although IBM-805 Test Scoring Machine was very effective, eventually, with time, its speed made it limited in terms of use.

In 1955, E.F. Lindquist developed the first genuine Optical Mark Reader, which recognized marked answers as dark ovals by passing light through answer sheets, and it could process 4000 sheets per hour, five times better than IBM-805 Test Scoring Machine [5].

In 1972, Scantron Corp. was founded by Michael Sokolski, which made it easier for K-12 schools and higher-ed institutes to use OMR technology by making smaller and less expensive OMR scanners. Scantron became so popular that their proprietary green and white coloured answer sheets were still selling for 15 to 20 cents per sheet in 2012 [6].

1.2.2 Software-based OMR

Hardware-based OMR solutions were expensive because they needed a dedicated scanner and specially designed sheets which cost around 10 cents per sheet on average. To counter that, in the early 1990s, OMR technology took the next giant leap by introducing software-based solutions. Gravic, Inc. released one of the first software solutions in 1991, called Remark Office OMR 1.0 [7].

In Software-based OMR solutions, consumers can use any readily available scanner to scan the sheet, which makes it very inexpensive compared to hardware-based solutions. The OMR task is divided into two parts: scanning the sheet using a scanner and identifying the markings from that sheet.

1.2.3 Webcam-based OMR

Patel et al. [8] categorized OMR technology into three generations. The first-generation OMR readers are hardware-based solutions that measure the reflectivity of light from the sheet at different places. The second-generation OMR readers are software-based solutions which use traditional scanners to scan OMR sheets and then apply image processing and computer vision techniques to tabulate the data. Finally, in the third-generation OMR readers, a mobile camera or a webcam is used to capture the OMR sheet, and then image processing techniques are applied to extract information.

The lower cost, ease of use, and faster speed than hardware and software solutions make webcam-based OMR a domain of active research interest and innovation. However, the camera's resolution, disorientation of the image, and variable lighting conditions make webcam-based solutions more error-prone than software-based solutions.

1.3 Technical Difficulties in Webcam-based OMR

There are some technical difficulties associated with webcam-based OMR solutions.

- Lighting Conditions – In real-life scenarios, not every user can ensure that the OMR system is used in ideal lighting conditions. Shadows in the dark and glares in the bright lights can cause the image of the OMR sheet to lose information.
- Resolution of Webcam – Low-resolution camera affects the quality of the captured image by making it look blurry and jagged because of fewer pixels.
- Image Skewness – Because the OMR system needs user input, it is impossible to predict whether the user will always hold the sheet in front of the webcam at an ideal angle.

1.4 Thesis Motivation

In academia, Multiple Choice Question (MCQs) is a basic and most frequently used method to objectively assess a student's understanding and knowledge of a particular subject. As MCQs do not need any subjective input from the student, automating their checking saves a lot of time for the examiner.

Researchers have tackled many different variants of automating the marking of exams. Numerous hardware-based and software-based solutions are designed to solve this problem. However, the main difference between the solutions is the cost and speed. Not every examiner can buy a dedicated scanner or expensive software that would take minutes to scan a sheet and give results.

Webcam-based OMR solutions offer an excellent advantage for many instructors worldwide, as many have limited resources. A good affordable webcam-based OMR can save a lot of time for instructors, especially in developing countries.

1.5 Problem Statement and Thesis Contributions

This thesis aims to design a webcam-based OMR system that can cover more real-life scenarios than existing implementations.

Each existing implementation has a set of limitations, making it challenging to use them from a new user's perspective. Some of them need to manually crop out the background of the paper and fix the skewness of the image, which eliminates the whole point of saving time. At the same time, others expect their users to use high-resolution (1080p or more) webcams, which are not available in most cheap or mid-range laptops.

This thesis aims to address the current issues in webcam-based OMR by making the following contributions:

- It covers many cheap and mid-range laptops by supporting resolutions from 720p and above.
- The only manual input it needs from the user is to capture the image of the OMR sheet and provide the correct answer key.
- The implementation is designed to minimize the time taken to do the processing and give optimal results.
- It will always give 100% accuracy; if that is not possible, it will show a warning message rather than providing incorrect results.
- It supports any orientation of the OMR sheet while capturing the image.

- It can also detect the roll number or ID number of the students, so the instructor has to make a minimal effort while assessing the tests.
- It supports several illumination levels, from a dull-lighted room to a room with daylight (sunny) coming in.

1.6 Thesis Outline

There are five chapters in this thesis, organized in the following manner:

- Chapter 1 describes OMR's history, thesis motivation, problem statement, and contributions.
- Chapter 2 reviews the background study of essential concepts in the image processing domain to better understand the topic.
- Chapter 3 covers the existing work that various researchers in this area have put in.
- Chapter 4 explains the proposed methodology of our webcam-based OMR.
- Chapter 5 discusses the experiments and results of the thesis and its comparison with existing implementations and algorithms.
- Chapter 6 concludes the thesis by summarizing the work and giving a direction for future work.

Chapter 2 – Background Study

This chapter covers a few topics required to understand this thesis's purpose and work.

2.1 Image Processing Terminology

It is essential to know about some of the terminologies that will be used throughout the thesis to get a better understanding of the subject.

Coloured Image: As shown in Figure 1, in a coloured image, each pixel of primary colours, red, blue, and green, is represented by a vector of three numbers (each number ranges from 0 to 255).

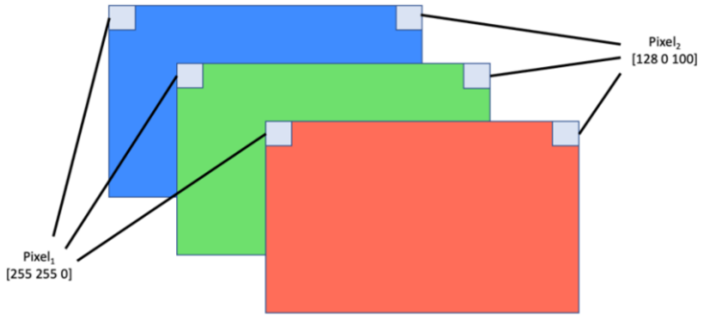


Figure 1: Example of a Coloured Image's Pixels

Grayscale Image: As shown in Figure 2, in grayscale images, each pixel represents the intensity information by a single number ranging from 0 (black) to 255 (white).

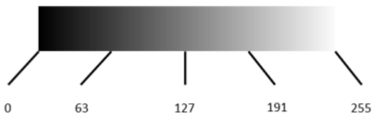


Figure 2: Intensity Information of each Gray Level

Binary Image: As shown in Figure 3, in a binary image, each pixel can be only of two colours, black and white.

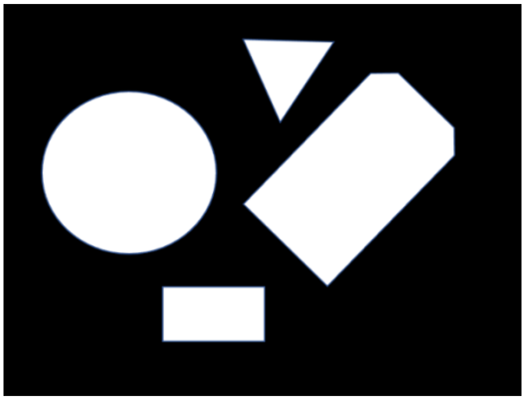


Figure 3: Example of a Binary Image

Image Binarization: Converting a grayscale image into a binary image. There are many image binarization techniques (explained ahead) divided into global and local categories. In global techniques, one value (called threshold value) is chosen from the whole image to create a binary image. While in local techniques, different threshold values are determined for each pixel based on the characteristics of their surrounding areas.

Blob Detection: A blob stands for Binary Large Object and can be described as a set of connected pixels inside a binary image. There are two main types of connectivity: 8-connectivity and 4-connectivity. The former differs from the latter by handling diagonal pixels, which makes it better in real-life scenarios. However, 4-connectivity is computationally cheaper than 8-connectivity. Figure 4 depicts the neighbourhood of both connectivity types.

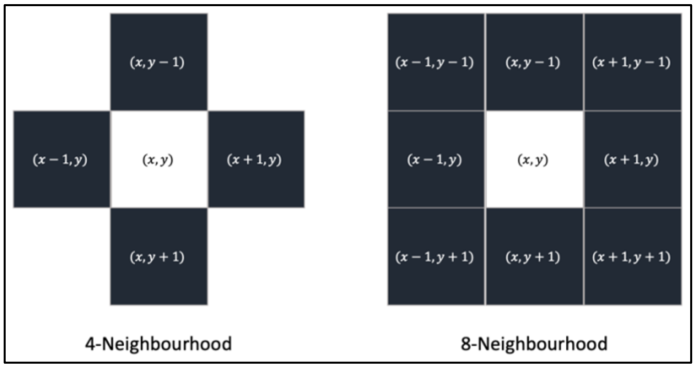


Figure 4: Pixel Connectivity Neighbourhoods

2.2 Image Binarization Algorithms

2.2.1 Simple Image Binarization

Simple Image Binarization, also called Simple Thresholding, is a global binarization technique in which one threshold value is chosen for the whole image. If the pixel value is smaller than that threshold, it is assigned a value of 0; otherwise, it is given the maximum value.

$$B(x,y) = \begin{cases} maxValue & if\ I(x,y) > t \\ 0 & otherwise \end{cases} \qquad (1)$$

Here, $B(x,y)$ is the resultant binarized image, t is the selected threshold value, and $I(x,y)$ is the intensity of the pixel at location (x,y).

2.2.2 Adaptive Image Binarization

Adaptive Image Binarization, also called Adaptive Thresholding, works by determining a threshold for each pixel based on its surrounding neighbours. This is useful in the case of varying lighting conditions across different parts of an image, as shown in Figure 5.

There are two variants of Adaptive Thresholding used in OpenCV to calculate the threshold value:

- ADAPTIVE_THRESH_MEAN_C: The threshold value is calculated by computing the mean of the pixel's neighbourhood (defined as $blockSize$), minus a constant C which is subtracted from the mean or weighted sum of the neighbourhood pixels.
- ADAPTIVE_THRESH_GAUSSIAN_C: The threshold value is calculated as a gaussian-weighted sum of the pixel's neighbourhood (defined as $blockSize$), minus a constant C.

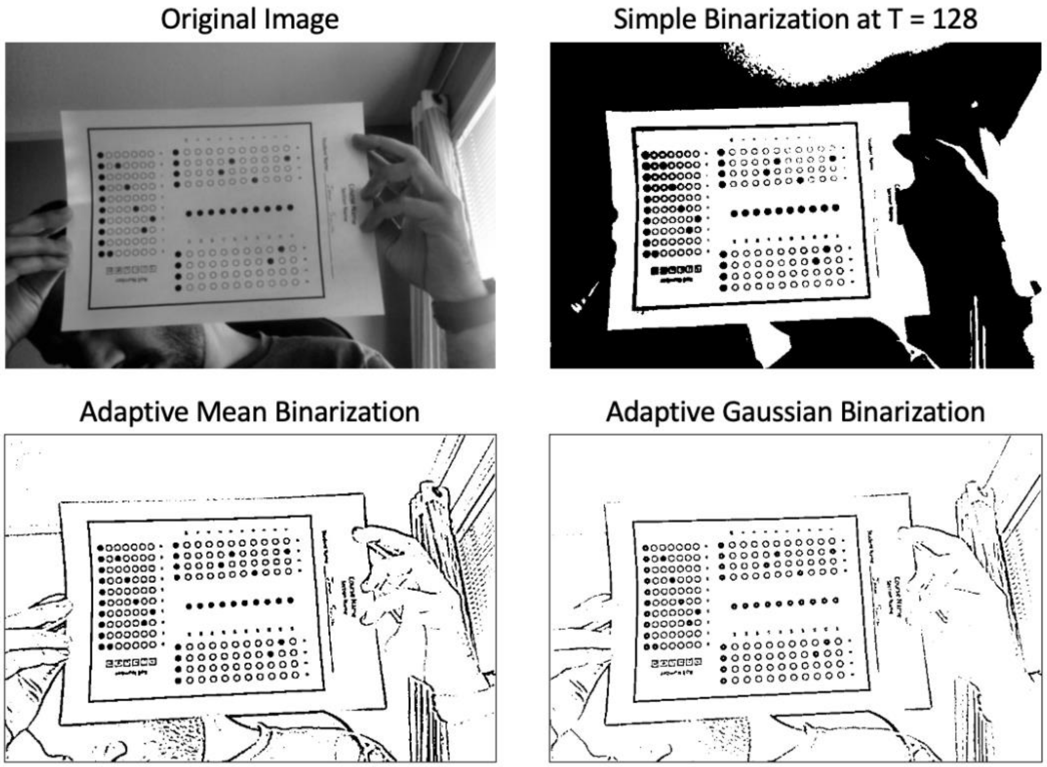

Original Image
Simple Binarization at T = 128
Adaptive Mean Binarization
Adaptive Gaussian Binarization

Figure 5: Example of Simple and Adaptive Binarization

2.2.3 Otsu Binarization

Otsu Binarization [9] is also a global binarization, but the threshold value is calculated automatically instead of choosing an arbitrary value. The main idea is to search for a threshold value that can maximize the inter-class variance or minimize the intra-class variance. So that the histogram of the image will have two distinct interchangeable classes of background and foreground, as shown in Figure 6. Otsu Binarization is a vital part of this thesis; more details about it are covered in Chapter 4.

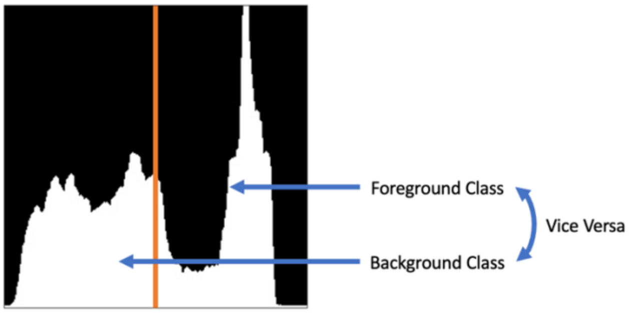


Figure 6: Otsu Binarization Algorithm

2.2.4 Niblack Binarization

Niblack [10] is a localized binarization algorithm in which the threshold value for each pixel $t(x, y)$ is calculated as the sum of the mean m of its $n \times n$ neighbourhood, and the standard deviation of those neighbourhood pixels σ, multiplied by a constant k.

$$t(x, y) = m + (k\sigma) \tag{2}$$

k has a default value of -0.2, which Niblack calculated as part of the experiments performed in [10].

2.2.5 Sauvola Binarization

Sauvola Binarization [11] is an improvement over Niblack binarization. In Sauvola, the dynamic range of standard deviation R is used, which increases the adaptability of standard deviation. For grayscale images, 128 is used as a value of R, and k has a default value between 0.2 and 0.5.

$$t(x, y) = m \times \left[1 - k\left(1 - \frac{\sigma}{R}\right)\right] \tag{3}$$

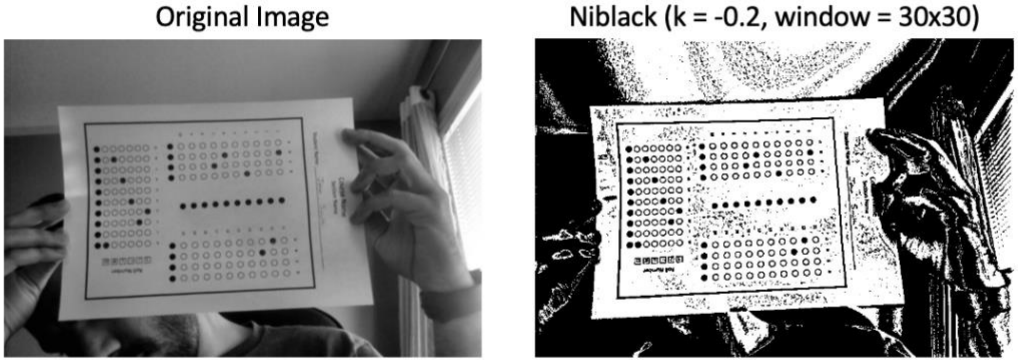
Niblack Binarization faces issues in regions with no contrast difference by introducing a lot of noise. Sauvola Binarization addresses this issue by taking $t(x, y)$ below m to remove darker regions, thus avoiding noise in the resultant binary image.

2.2.6 Wolf & Jolion Binarization

Wolf et al. Binarization [12] further improve on Sauvola Binarization by setting the value of R to the maximum standard deviations of all the neighbourhood windows and introducing a factor of M, which is the minimum gray level of the image.

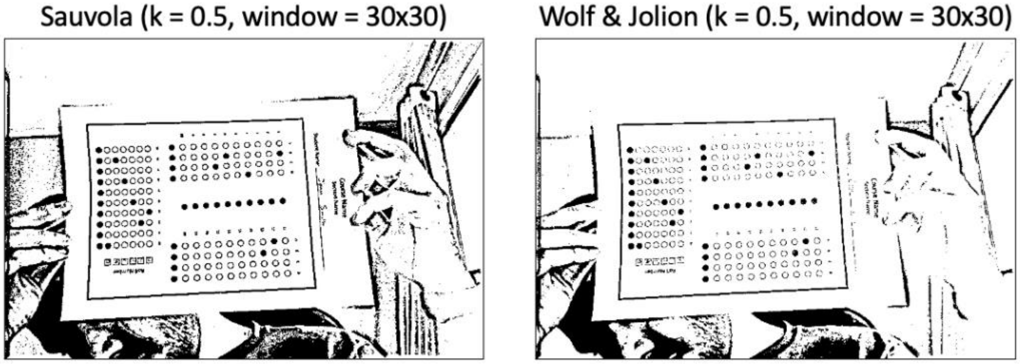
$$t(x, y) = m - k(1 - \frac{\sigma}{R})(m - M) \tag{4}$$

Figure 7 shows the binarization output of Niblack, Sauvola, and Wolf & Jolion Binarization algorithms. As evident from the figure, Wolf & Jolion Binarization performs better than Sauvola Binarization, which serves better than Niblack Binarization based on lesser noise and better foreground and background separation.



Original Image

Niblack (k = -0.2, window = 30x30)



Sauvola (k = 0.5, window = 30x30)

Wolf & Jolion (k = 0.5, window = 30x30)

Figure 7: Example of Niblack, Sauvola, Wolf & Jolion Binarization Algorithms

2.3 OpenCV and its algorithms

OpenCV (Open-Source Computer Vision Library) [13] is an open-source, cross-platform library used for image and video processing. It contains around 2500 algorithms that facilitate its user to perform computer vision and machine learning tasks.

Some of the applications of OpenCV are detecting and recognizing faces, classifying human gestures and activities, extracting 3D models of objects, concatenating low-resolution images to produce a high-quality picture, motion tracking, augmented reality etc.

OpenCV works with many leading programming languages, such as C++, Java, Python etc. It was launched in the early 2000s, so several learning resources are available for the public to learn and use OpenCV. This research work has OpenCV at its core; however, other excellent libraries, such as TensorFlow, PyTorch, MATLAB etc., can perform similar tasks.

2.3.1 SimpleBlobDetector

OpenCV's SimpleBlobDetector [14] algorithm detects blobs inside the image based on different features. The working of this algorithm is based on the following steps:

- First, the input image is converted into several binary images. A different threshold value is chosen for each binary image. Starting from $minThreshold$ (inclusive), until $maxThreshold$ (exclusive) by incrementing with $thresholdStep$. Hence, the first threshold is $minThreshold$, and the second threshold is $minThreshold + stepThreshold$, while the third threshold is $minThreshold + 2 \times thresholdStep$, and so on.
- Then, connected components are grouped inside each binary image.
- In the third step, the centres of all the connected components are computed, and those with a distance less than a controlled parameter, called $minDistBetweenBlobs$, are merged.

- Finally, centres of newly merged blobs and their radiuses are returned.

2.3.2 FindContours

OpenCV's FindContours [15] algorithm finds contours inside an image. A contour is a curve formed by joining all the points having the same colour or intensity along the boundary of an object. They are used for object detection and recognition, shape analysis etc.

In OpenCV's implementation of finding contours, the goal is to find a white object from a black background. This is helpful information because the OMR sheet is a paper of white colour, and the background is always darker than that white paper.

There are two methods to approximate the contours. Figure 8 shows the difference in working between the two techniques.

- CHAIN_APPROX_NONE: All the boundary points along a region are stored in this contour approximation technique.
- CHAIN_APPROX_SIMPLE: It ignores the vertical, horizontal, and diagonal points along the region and only stores the endpoints. This makes it faster than the other method.

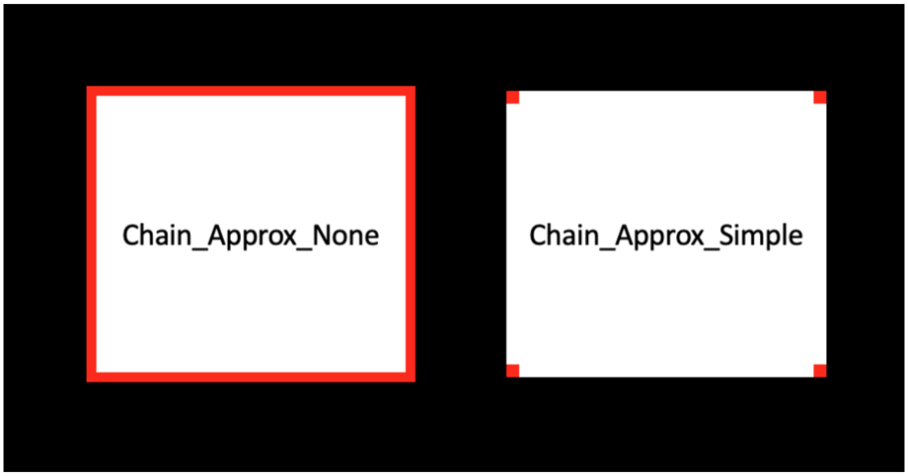


Figure 8: Example of OpenCV's FindContours Approximation Methods

2.3.3 MinAreaRect

OpenCV's MinAreaRect [16] algorithm simply finds a rotated rectangle when given a set of 2D points. As the name suggests, the area of that rotated rectangle is the minimum possible. In this thesis, after finding contours and getting a set of 2D points, this function is used to plot the boundary rectangle for the OMR sheet.

In figure 9, an arrow-shaped object has green as a rotated boundary box with minimal area, while red is a regular bounding rectangle.

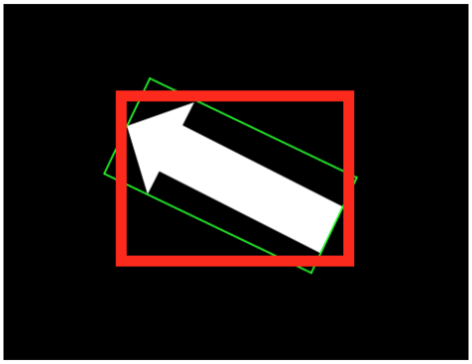


Figure 9: Example of OpenCV's MinAreaRect

Chapter 3 – Related Work

This chapter includes related work that motivated our proposed approach. Every research on webcam-based OMR has offered a different solution to the problem. Some used different kinds of image processing techniques, while others used machine learning algorithms to get good accuracy.

For this thesis, we studied many approaches, including work on hardware-based OMRs and software-based OMRs. However, this section will only cover third-generation OMRs [8] to keep it related to the topic.

3.1 Paper 1 – Shape Matching OMR

In 2015, Talib et al. [17] proposed an approach to pattern recognition using shape-based template matching. Their algorithm consisted of two phases: the training phase and the recognition phase, as evident from Figure 10.

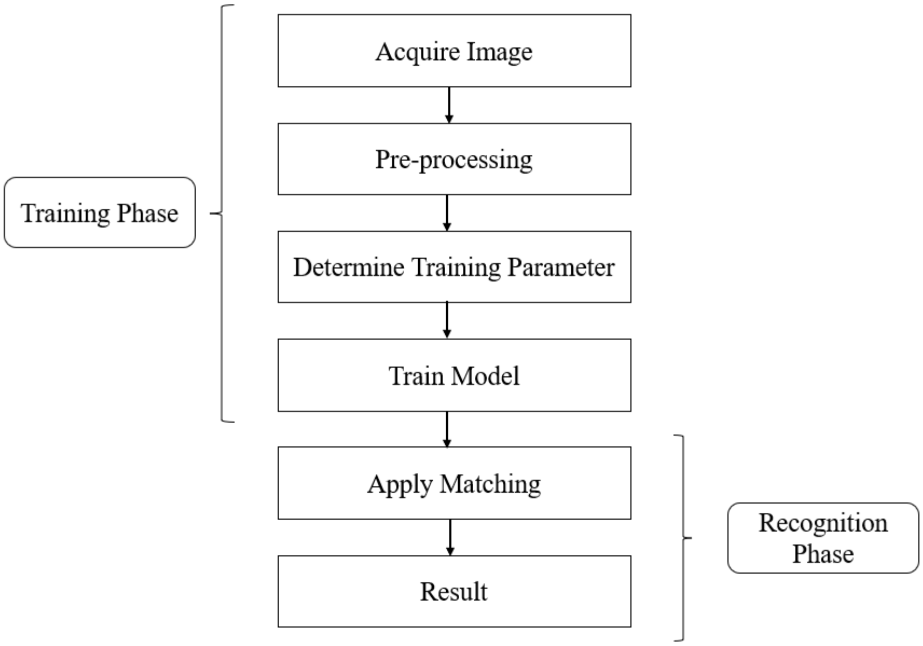


Figure 10: Shape Matching OMR

Training Phase: A high-contrast image from the webcam is taken, which is then pre-processed by converting it into a grayscale image and applying a mean filter to remove grain noise. Then, the region of interest (ROI) is selected manually by the user, in which all option bubbles and question numbers are chosen. After that, image thresholding is applied to minimize the effect of varying colour contrast and illumination changes. Then, the model is trained based on the selected ROI.

Recognition Phase: The search image is compared with the template image by matching the intensity values. A matching score is assigned to the search image based on the similarity between the candidate and template images. The score is based on measuring how many model points from the template image can be mapped onto the search image.

Then, image pyramids are generated to fasten the process. This helps reduce the overall working area on pixels and lessens the computational cost of the whole process.

Results: Based on a minimum of 80% similarity score, the model can achieve an accuracy of around 97.5%.

Constraints: There are a few constraints in this approach which restrict it from being used freely in real-life scenarios:

- The user must manually select the region of Interest (ROI) on the template image.
- A high-contrast search image is required to be able to omit a good accuracy score.
- The lighting must be constant throughout the whole working of the system, including training and recognition.

3.2 Paper 2 – Template Matching OMR

In 2015, Karunanayake [18] proposed a similar approach to that of Talib et al. but only using image processing techniques. The main steps of the image processing techniques are shown in Figure 11. First, a template image is captured with a plain and dark background to separate ROI. After marking the largest rectangular area as ROI, the selected region is straightened out and stored in a matrix.

Then, the search images are captured without changing the camera's height and distance configuration. Like the template image, search images are also saved in a matrix. In the template matching phase, each search image is matched with the template image based on a numerical index so that the two images are aligned with each other. The search image $I_{M \times N}$, the template image $T_{m \times n}$, and the resultant numerical index $R_{(M-m+1,N-n+1)}$ are computed as:

$$R_{(x,y)} = \frac{R_{(sq-diff)(x,y)}}{Z_{(x,y)}} \tag{5}$$

$$R_{(sq-diff)} = \sum_{(x',y')} [T(x',y') - I(x+x',y+y')]^2 \tag{6}$$

$$x' = 0,1,\ldots,m-1 \; and \; y' = 0,1,\ldots,n-1$$

$$Z_{(x,y)} = \sqrt{\sum_{(x',y')} T(x',y')^2 \times \sum_{(x',y')} I(x+x',y+y')^2} \tag{7}$$

After the template and search images are aligned, they are converted into binary images to only show filled-out options. As the template image's options are correctly filled, the template binary image is 'AND' with the search binary image to eliminate the incorrect choices. Then, anything smaller than 80 pixels is removed from the image so that lesser noise is present. Finally, the remaining white regions are counted as the correct selected options to compile results.

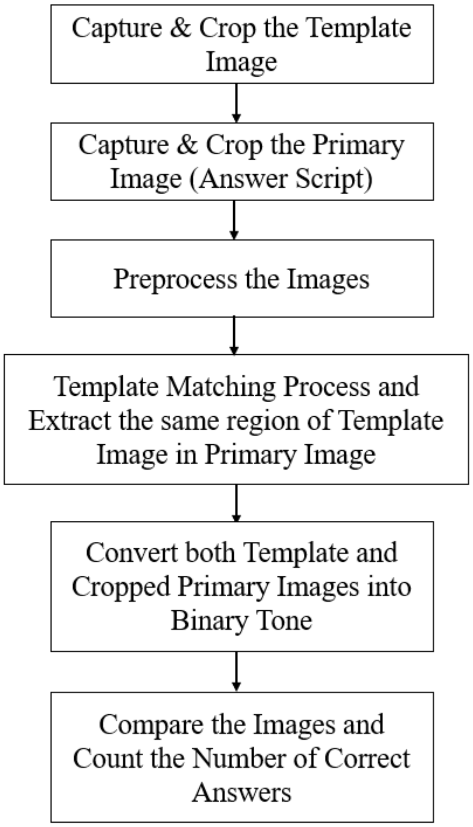

Capture & Crop the Template Image
Capture & Crop the Primary Image (Answer Script)
Preprocess the Images
Template Matching Process and Extract the same region of Template Image in Primary Image
Convert both Template and Cropped Primary Images into Binary Tone
Compare the Images and Count the Number of Correct Answers

Figure 11: Template Matching OMR

Results: The accuracy of experimentation on three types of answer sheets is 97.6%. The average reported time for each sheet processing is less than 2 seconds.

Constraints: There are a few constraints in this approach which restrict it from being used freely in real-life scenarios:

- The ROI will not be cropped out if a high contrast does not exist between the image and the background.
- Between different answer sheets, the camera cannot move; otherwise, the template image must be captured again.
- The lighting must be constant over the whole sheet.

3.3 Paper 3 – Mobile Camera OMR

In 2019, Rasiq et al. [19] proposed an approach in which the image is captured using a mobile phone camera. The algorithm uses the total number of questions and columns to generate an OMR sheet. Then, filled OMR sheets are scanned via mobile camera and the answer area is extracted. Then, for each question, all options are separated. Finally, black pixels are counted for each separated option. Figure 12 shows a flowchart diagram of the complete working of the algorithm.

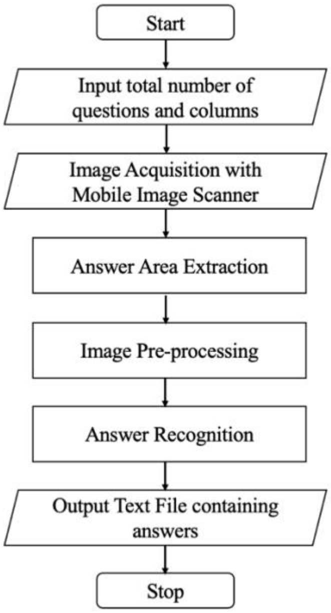

Start
Input total number of questions and columns
Image Acquisition with Mobile Image Scanner
Answer Area Extraction
Image Pre-processing
Answer Recognition
Output Text File containing answers
Stop

Figure 12: Flowchart of Mobile-based OMR

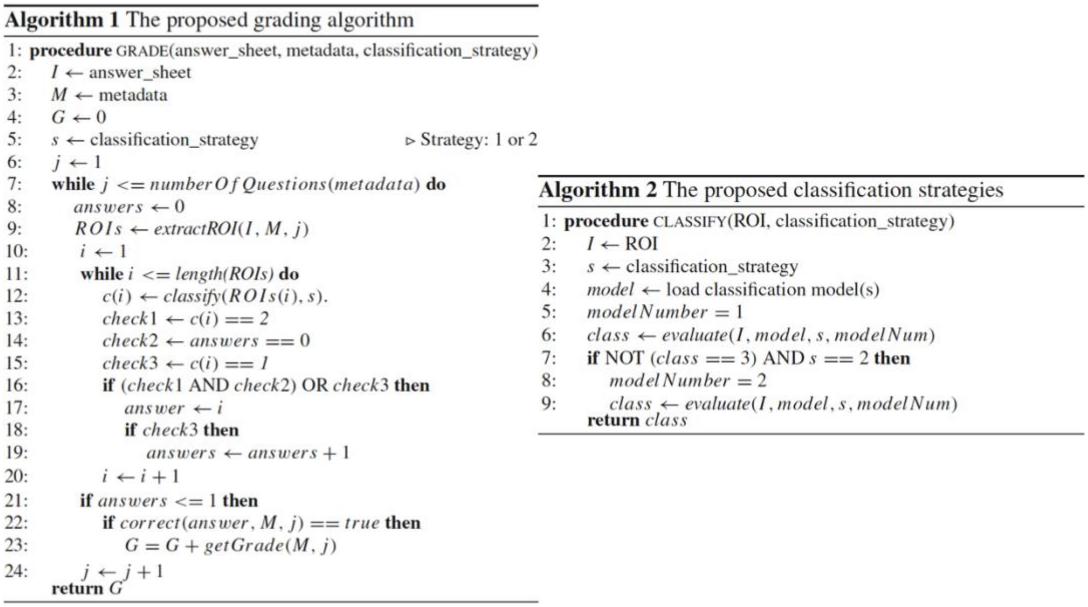
Results: The best possible accuracy of the algorithm, when the option circle is at least 45% filled, is 99.44%.

Constraints: Few constraints in this approach are:

- Mobile Phone Camera is not very ideal for instructors to use. As they want to keep their data on their computer in most cases and transferring files from mobile to computer for each student assessment is not at all timesaving.
- The accuracy was only calculated on five images, with no information about the total time taken.

3.4 Paper 4 – Machine Learning OMR

In 2019, Afifi et al. [20] tried to solve the problem with machine learning classifiers. Their approach involves the scanning of the answer sheet using a scanner. The answer sheet must be aligned with the model answer sheet to extract ROIs for each question separately. Then, each ROI is classified into three classes: confirmed, crossed out and an empty box. The classifiers used in the experimentations are the Naïve Bayes Classifier (NBC), Bag of Visual Words (BoVW), and Convolutional Neural Network (CNN). Figure 13 taken from [20] shows the algorithms that grade answer sheets (left) and select classification strategies (right).



Algorithm 1 The proposed grading algorithm

```
procedure GRADE(answer_sheet, metadata, classification_strategy)
    $I \leftarrow$ answer_sheet
    $M \leftarrow$ metadata
    $G \leftarrow 0$
    $s \leftarrow$ classification_strategy                ▷ Strategy: 1 or 2
    $j \leftarrow 1$
    while $j <= numberOfQuestions(metadata)$ do
        $answers \leftarrow 0$
        $ROIs \leftarrow extractROI(I, M, j)$
        $i \leftarrow 1$
        while $i <= length(ROIs)$ do
            $c(i) \leftarrow classify(ROIs(i), s)$.
            $check1 \leftarrow c(i) == 2$
            $check2 \leftarrow answers == 0$
            $check3 \leftarrow c(i) == 1$
            if ($check1$ AND $check2$) OR $check3$ then
                $answer \leftarrow i$
                if $check3$ then
                    $answers \leftarrow answers + 1$
            $i \leftarrow i + 1$
        if $answers <= 1$ then
            if $correct(answer, M, j) == true$ then
                $G = G + getGrade(M, j)$
        $j \leftarrow j + 1$
    return $G$
```

Algorithm 2 The proposed classification strategies

```
procedure CLASSIFY(ROI, classification_strategy)
    $I \leftarrow$ ROI
    $s \leftarrow$ classification_strategy
    $model \leftarrow$ load classification model(s)
    $modelNumber = 1$
    $class \leftarrow evaluate(I, model, s, modelNum)$
    if NOT ($class == 3$) AND $s == 2$ then
        $modelNumber = 2$
        $class \leftarrow evaluate(I, model, s, modelNum)$
    return $class$
```

Figure 13: Algorithms used in Machine Learning OMR

Results: CNN gave the best accuracy of 99.39% on questions (correct questions / total questions), while the accuracy on answer sheets (answer sheets properly marked / total answer sheet) was 94.02%.

Constraints: The primary limitation of this approach was the use of a scanner to align answer sheets with a model answer sheet.

3.5 Paper 5 – Enhanced Algorithm for OMR

In 2021, Jingyi et al. [21] proposed an algorithm consisting of several image-processing techniques. First, a template image is captured using a camera, having distinct darker background. The template image tells the system the total number of questions and options per question.

The template image is then converted to grayscale, and Otsu Threshold is applied to generate a binary image. Then, the pixels are dilated to fill the gaps in the broken pixels. Then, contours are applied to the binary image to detect bubbles with the same aspect ratio. Then, indexes of all the detected bubbles are stored in a dictionary, and the same procedure is performed for the student's answer sheet image.

After creating the two dictionaries, they are compared with one another. If the index number for a question in the search dictionary matches the template dictionary, then that bubble is correctly filled and considered valid.

Results: For a result set of 120 questions and 600 bubbles, the proposed algorithm has an F1-score of 97.72% because of the over-detection of darkened bubbles in a scenario.

Constraints: Some limitations of this approach are:

- Mobile Phone Camera is required to capture the images. As stated earlier, this is not ideal for instructors and cannot be considered a time-saver.
- The system's overall accuracy can be damaged if the background and the OMR sheet are not distinguishable enough. The background must be very dark for the system to extract ROI properly.

3.6 Paper 6 – Three-Scaled Images OMR

In 2022, Raveendran [22] proposed a technique for webcam-based OMR by taking three variants of an input image to get an ideal distance if the image is far from the camera. Then, each scaled image is divided into 11x11 patches, and the mean variance of each patch is calculated. Images are binarized based on the mean-variance from each patch.

Then, a canny edge detector is applied to find the edges of all blobs inside the binary image. Inter-blob distance is computed to separate the reference blobs. The reference points are then connected horizontally and vertically. Then, the centroids from the top-bottom and left-right reference blobs are calculated to validate the orientation of the image. Finally, the filled circles along the reference points are sorted and compiled into an output. Figure 14 portrays the block diagram of the proposed methodology.

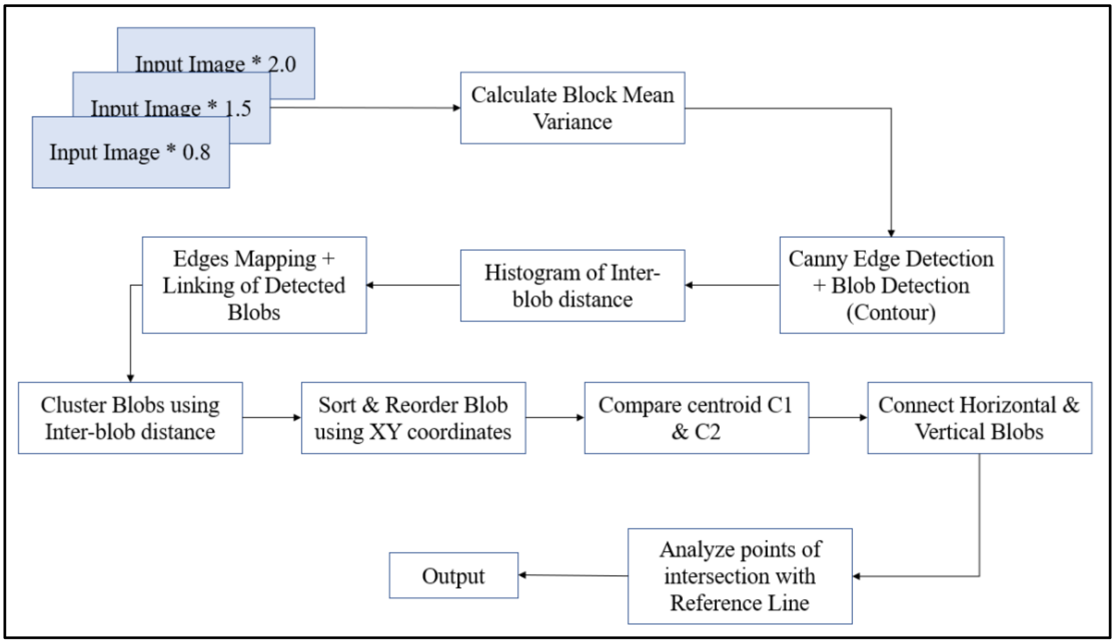

Input Image * 2.0
Input Image * 1.5
Input Image * 0.8
Calculate Block Mean Variance
Canny Edge Detection + Blob Detection (Contour)
Histogram of Inter-blob distance
Edges Mapping + Linking of Detected Blobs
Cluster Blobs using Inter-blob distance
Sort & Reorder Blob using XY coordinates
Compare centroid C1 & C2
Connect Horizontal & Vertical Blobs
Analyze points of intersection with Reference Line
Output

Figure 14: Block Diagram of Three-Scaled Images OMR

Results: 100% accuracy is achieved with this method. When the system cannot detect reference blob points, the user is shown a warning message to correct the OMR sheet's orientation.

Constraints: Some limitations of this approach are:

- 1080p webcam is used, which eliminates many cheap and mid-range computers as the supported environments.
- With lighting changes and shadows, results are unexpected, with many warning messages.

Chapter 4 – Methodology

This chapter discusses our algorithm for a webcam-based OMR solution in detail. A summary of all image-processing steps involved in the methodology is illustrated in Figure 15. There are three main stages in the working of the algorithm, Pre-Processing, ROI Extraction and Blob Detection.

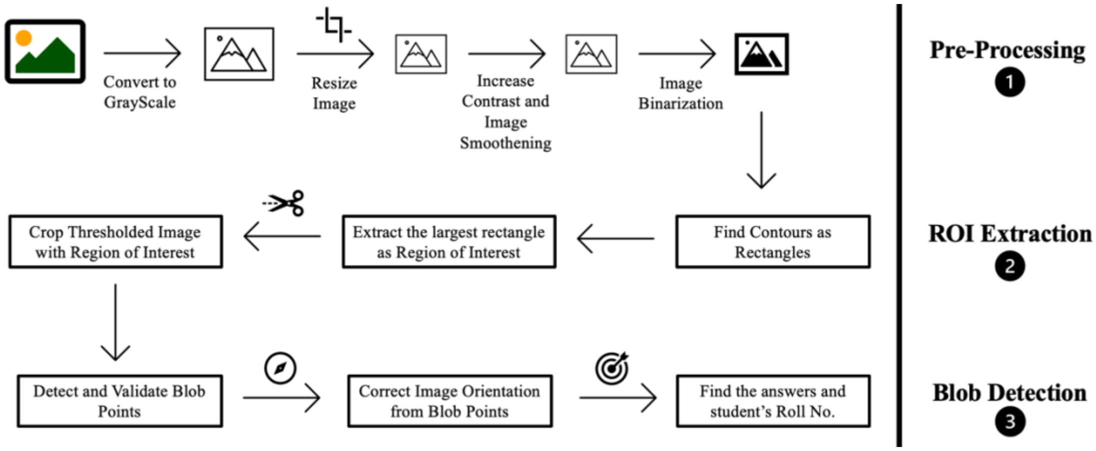

Convert to GrayScale
Resize Image
Increase Contrast and Image Smoothening
Image Binarization
Pre-Processing
1
Find Contours as Rectangles
Extract the largest rectangle as Region of Interest
Crop Thresholded Image with Region of Interest
ROI Extraction
2
Detect and Validate Blob Points
Correct Image Orientation from Blob Points
Find the answers and student's Roll No.
Blob Detection
3

Figure 15: Proposed Webcam-OMR System

4.1 Stage 1 – Pre-Processing

In the Pre-Processing stage, well-known image processing techniques are employed to obtain a binary image. First, a specifically designed OMR sheet for this system is filled and captured using a webcam without any filters. Then, the image is fed to the system and read as a three-channel BGR (Blue, Green, Red) image.

4.1.1 BGR to Grayscale

The BGR image is then converted to a grayscale image $I_{Grayscale}$ by multiplying each channel of RGB with specific constants.

$$I_{Grayscale} = 0.299 \times R + 0.587 \times G + 0.114 \times B \tag{8}$$

Figure 16 shows the initial BGR image (left), which is then converted into a grayscale image (right).

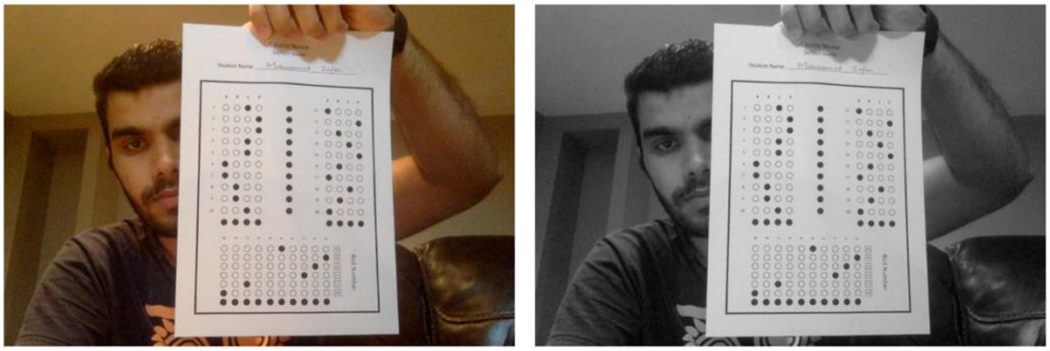


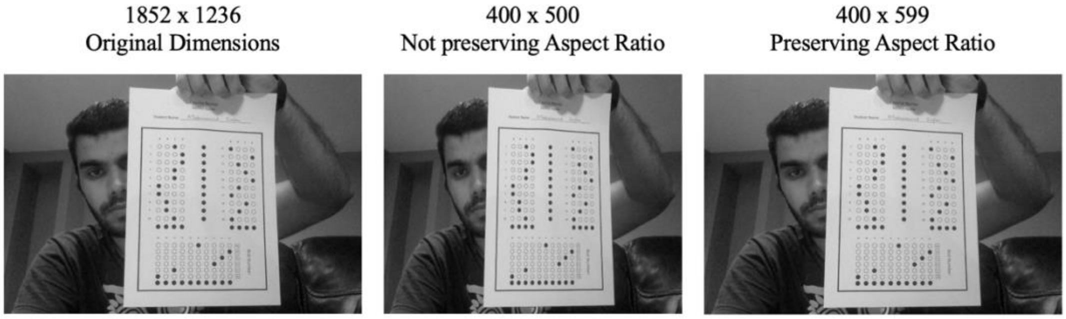
Figure 16: Conversion of a BGR Image to a Grayscale Image

4.1.2 Grayscale Image Resizing

The size of the grayscale image, represented as $M_1 \times N_1$, is then changed to a smaller size, defined as $M_2 \times N_2$, so that the system will have lesser pixels to work on. This increases the system's overall performance by dramatically reducing the computational cost. The height N_2 of the resized image $I_{resized}$ is set to 400 pixels, and the width is calculated as:

$$M_2 = M_1 \times \frac{N_2}{N_1} \tag{9}$$

This helps preserve the aspect ratio of the image so that the proportionality between the height and the width does not affect the composure of the image. Figure 17 demonstrates how the outlook of the image is changed if the aspect ratio is not preserved.


1852 x 1236
Original Dimensions

400 x 500
Not preserving Aspect Ratio

400 x 599
Preserving Aspect Ratio

Figure 17: Example of Aspect Ratio Importance

4.1.3 Contrast Enhancement

In realistic scenarios, images can take many forms of illumination. Imagine a case in which almost all the image is bright or nearly all the image is dark. In such cases, if the grayscale histogram is plotted, the intensity distribution of pixels will be skewed towards the right for bright images, and it will be skewed towards the left for dark images.

Figure 18 shows how by stretching the peaks inside the image, overall contrast can be increased. This process is called Histogram Equalization.

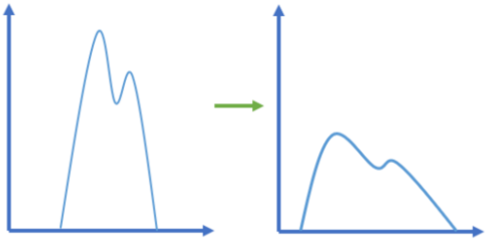


Figure 18: Example of Histogram Equalization

Contrast Limited Adaptive Histogram Equalization (CLAHE) is used for our system as it performs better in dynamic environments. The image is divided into several (x by x) patches, then each patch is histogram equalized. And, if a patch has more pixel distribution than a set contrast limit, the overhead is trimmed and distributed over all the other patches.

Figure 19 tells the difference CLAHE makes on our image from the last step. On the left, it shows resized grayscale image, while on the right, it has a contrast-enhanced image.

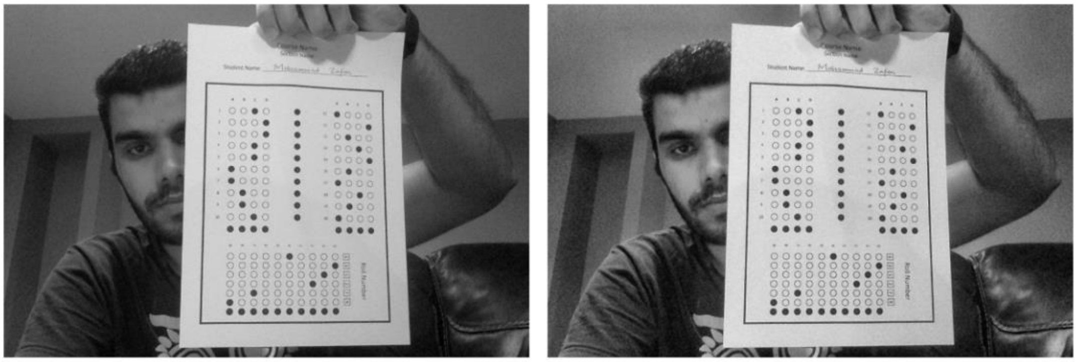


Figure 19: Example of CLAHE Effect

4.1.4 Image Smoothening

Next, the image is smoothened out by applying a gaussian filter. This minimizes the grain noise inside the image and removes unimportant edges, thus helping suppress false contours (discussed later).

Image Smoothening (also called image blurring or image filtering) is achieved by convolving a filter on the image. The filter we used for our work is a 5x5 Gaussian Filter. The choice of a 5x5 Gaussian Filter instead of a 3x3, or 7x7, and so on, is based on experimentation. 3x3 Gaussian Filter does not do an excellent job of removing noise, making it not very useful for our case. On the other hand, 7x7 or 9x9 Gaussian Filters do not add any more value to the overall noise removal, yet they take more processing time.

Figure 20 exhibits the effect of a 5x5 gaussian filtering (right) on the contrast-enhanced image (left).

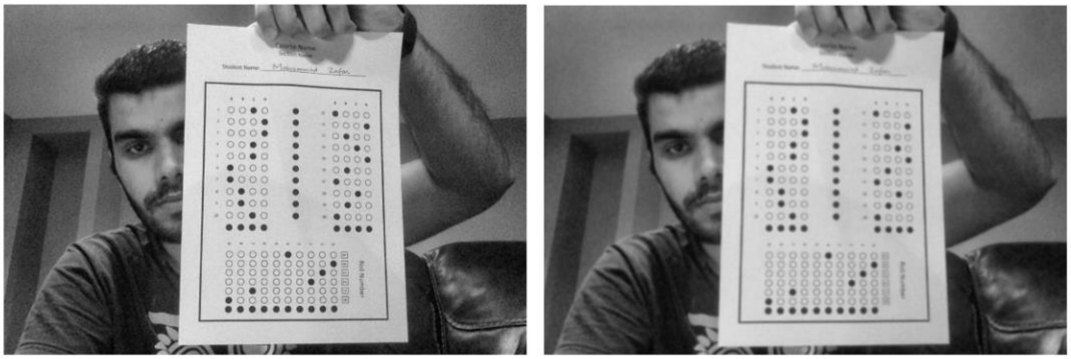


Figure 20: Example of 5x5 Gaussian Filter Effect

4.1.5 Image Binarization

Image Binarization is the most crucial step in our work, as it involves the integration of two algorithms to make them perform well for our specific task. In our case, image binarization aims to extract the region of interest from the background. As discussed in Chapter 2, the algorithms that require a constant value are unsuitable for our dynamic setting. Therefore, we used Otsu Binarization as our main algorithm to find the threshold value. Further, we discuss in depth the Otsu Binarization and another approach along the same lines by Quan et al. [23].

4.1.5.1 Otsu Binarization Detail

Otsu Binarization works on choosing a threshold value that perfectly separates the foreground from the background. However, perfect separation is impossible in real-life scenarios as a high degree of overlap exists between the two classes.

The key principle of Otsu Binarization is to maximize the inter-class variance or minimize the intra-class variance between the two classes: foreground and background. Variance simply measures the spread of the data. If the data points are more dispersed, then the variance will be high; otherwise, it will be low.

The Otsu Binarization algorithm works by iteratively finding a threshold that maximizes the inter-class variance, defined as:

$$\sigma^2_{inter-class}(t) = \omega_{bg}(t) \times \left(\mu_{Total} - \mu_{bg}(t)\right) + \omega_{fg}(t) \times \left(\mu_{Total} - \mu_{fg}(t)\right) \tag{10}$$

Here, $\omega_{bg}(t)\ and\ \omega_{fg}(t)$ represents the probability of the number of pixels at threshold t for background and foreground, respectively. $\mu_{bg}(t)\ and\ \mu_{fg}(t)$ represents the mean of both classes and μ_{Total} represents the total mean. The maximum intensity of the image is given as L. These values can be defined as:

$$\omega_{bg}(t) = \sum_{i=0}^{t} P(i) \qquad \omega_{fg}(t) = \sum_{i=t+1}^{L-1} P(i)$$

$$\mu_{bg}(t) = \frac{\sum_{i=0}^{t} iP(i)}{\omega_{bg}(t)} \qquad \mu_{fg}(t) = \frac{\sum_{i=t+1}^{L-1} iP(i)}{\omega_{fg}(t)}$$

$$P(i) = \frac{Number\ of\ pixels\ at\ i}{Total\ number\ of\ pixels} \qquad \mu_{Total} = \sum_{i=0}^{L-1} iP(i)$$

Figure 21 provides examples of Otsu Binarization. The Otsu Binarization works well in the binary image on top because the pixel intensity distribution is normal and is not skewed towards the darker or brighter regions. There is a certain degree of separation between the peak and neighbouring high-density intensity levels, as evident from its histogram. The optimal threshold value found by the Otsu Binarization algorithm is 137 for this image.

However, in the second binary image, the pixels are skewed towards the left (i.e., darker image), making it difficult for the algorithm to separate into two classes. It gives a smaller threshold value, making the filled points challenging to distinguish from other circles. These results show that the Otsu Binarization algorithm cannot be relied upon in a dynamic environment, especially with brighter and darker images.

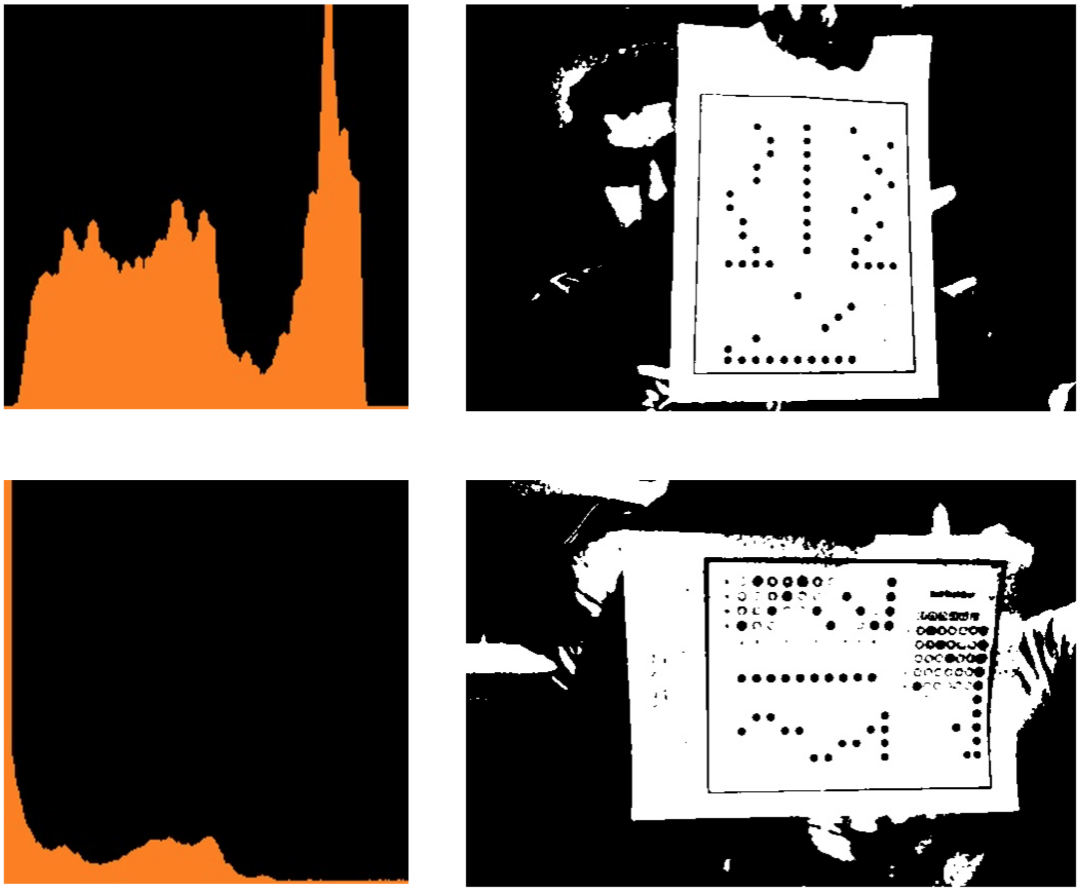


Figure 21: Examples of Otsu Binarization

4.1.5.2 Improved Otsu Binarization by Quan et al.

In 2019, Quan et al. [23] built on top of Otsu Binarization by introducing the neighbourhood pixel distribution into the calculation of an inter-class variance. In their specific problem, they mentioned that applying the neighbourhood pixel distribution on the background further darkens the image.

The inter-class variance with the neighbourhood is defined as:

$$\sigma^2_{inter-class}(t) = [\nu(t)] \times \left[\omega_{bg}(t) \times \left(\mu_{Total} - \mu_{bg}(t)\right)\right] + [1 - \nu(t)] \times \left[\omega_{fg}(t) \times \left(\mu_{Total} - \mu_{fg}(t)\right)\right] \tag{11}$$

Here:

$$\nu(t) = P(t-2) + P(t-1) + P(t) + P(t+1) + P(t+2)$$

$$P(t) = \frac{Number\ of\ pixels\ at\ t}{Total\ number\ of\ pixels}$$

4.1.5.3 Proposed Improvement of Otsu Binarization

As explained previously, the Otsu Binarization algorithm faces difficulties in darker and brighter OMR images. Then, we explored the findings from the work of Quan et al., which suggests that the introduction of neighbourhood pixel distribution on the background class of the image increases the inter-class variance [23].

The results from [23] mean that the resultant binary image appears darker, i.e., favouring the background. Building on this finding, it is easy to see that applying neighbourhood pixel distribution on the foreground class of the image will produce a binary image that appears brighter, i.e., favouring the foreground. Therefore, the brighter foreground can be found as:

$$\sigma^2_{inter-class}(t) = [1 - \nu(t)] \times \left[\omega_{bg}(t) \times \left(\mu_{Total} - \mu_{bg}(t)\right)\right] + [\nu(t)] \times \left[\omega_{fg}(t) \times \left(\mu_{Total} - \mu_{fg}(t)\right)\right] \tag{12}$$

Figure 22 shows the effects of these changes on an image. The binary image on the left shows the result with a brighter foreground. The binary image in the middle shows the result with normal Otsu Binarization. The binary image on the right shows the result with a darker background.

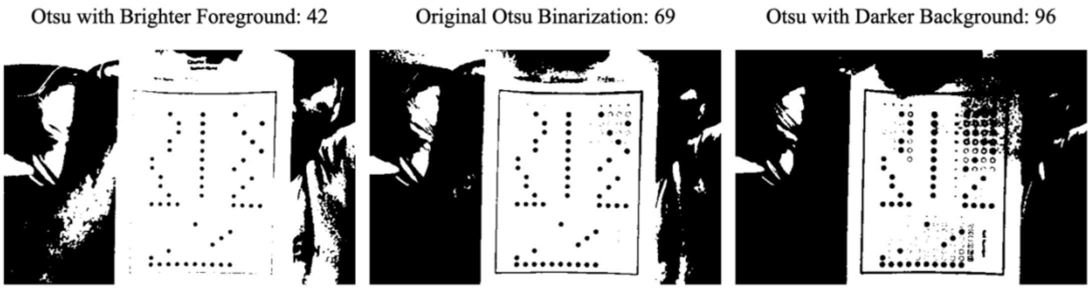

Otsu with Brighter Foreground: 42
Original Otsu Binarization: 69
Otsu with Darker Background: 96

Figure 22: Examples of different Otsu variants

The Otsu with brighter foreground diminishes details about the ROI. On the other hand, the Otsu with darker background adds too much noise to the binary image. That is why a factor must be added to the Otsu with brighter foreground to increase the black pixels or subtracted from the Otsu with a darker background to increase the white pixels.

By taking the mean between the Otsu Binarization and the Otsu with brighter foreground, we get a threshold value that solves our problem of diminishing ROI. And by taking the mean between the Otsu Binarization and the Otsu with a darker background, we get a threshold value that fixes the noise in the resultant image. To summarize, we will have the following three scenarios:

When the image is very dark: $threshold = \left\lfloor \frac{t_{bf} + t_O}{2} \right\rfloor$

When the image is very bright: $threshold = \left\lfloor \frac{t_{db} + t_O}{2} \right\rfloor$

When the image is in normal lighting conditions: $threshold = t_O$

Here, t_O is the Original Otsu Binarization as seen in (10), t_{bf} is the Otsu with Brighter Foreground as seen in (12), and t_{db} is the Otsu with Darker Background as seen in (11).

Whether an image is very dark or very bright is judged based on the total mean of the background and foreground classes inside the image, computed on L (maximum intensity) bins.

$$\mu_{Total} = \sum_{i=0}^{L-1} i \times \frac{Number\ of\ pixels\ at\ i}{Total\ number\ of\ pixels}$$

Based on experimentation, if $\mu_{Total} \leq 0.2 \times L$, it is considered as dark, and if $\mu_{Total} \geq 0.8 \times L$, it is considered as bright. Figure 23 shows the example of two images with their μ_{Total} computed.

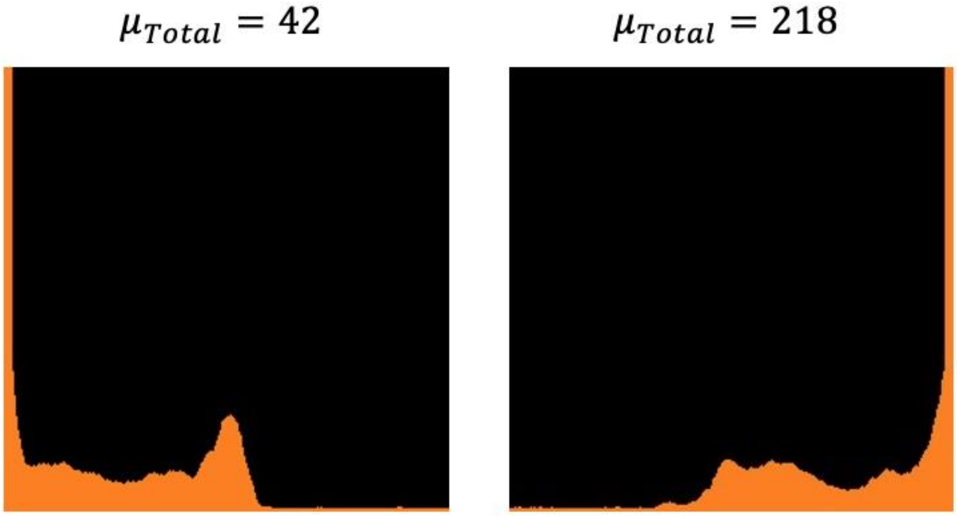


Figure 23: Total Mean computed on Dark and Bright Images

4.2 Stage 2 – ROI Extraction

Stage 2 is where the region of interest is extracted from the binary image obtained from the previous step.

4.2.1 Finding Contours & making rectangles

The contours in the binary image are found using OpenCV's *findContours* function. CHAIN_APPROX_SIMPLE technique is used to approximate the points of the contours. Then, using OpenCV's *minAreaRect* function, rectangles are drawn over all the detected contours. The rectangle with the largest parameter than the other rectangles is considered the Region of Interest.

Figure 24 shows an example of a binary image and the detected rectangular contours. In this example, ROI was ideally identified as the largest rectangle of the others.

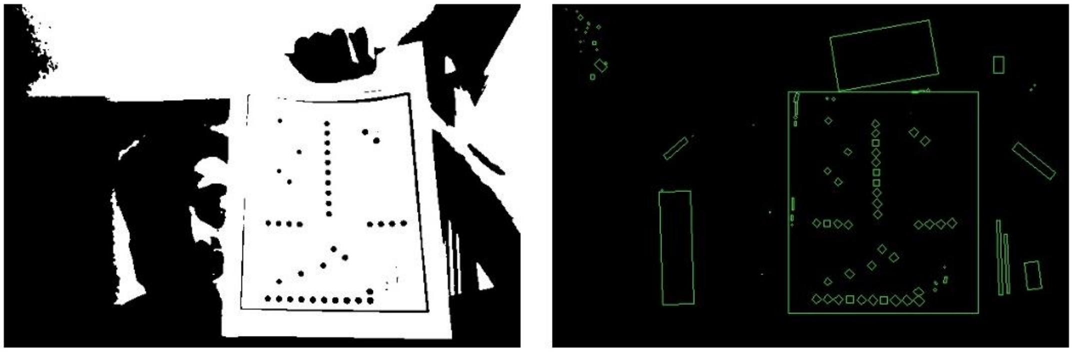


Figure 24: Example of Rectangular Contours

Sometimes it is possible that the edges of the paper are also detected as a rectangle, or the edges of the whole image are detected as a rectangle. These cases happen because of the same colour or intensity on those edges, so this is quite a realistic scenario. Figure 25 shows an example where the edges of the paper also make a rectangle.

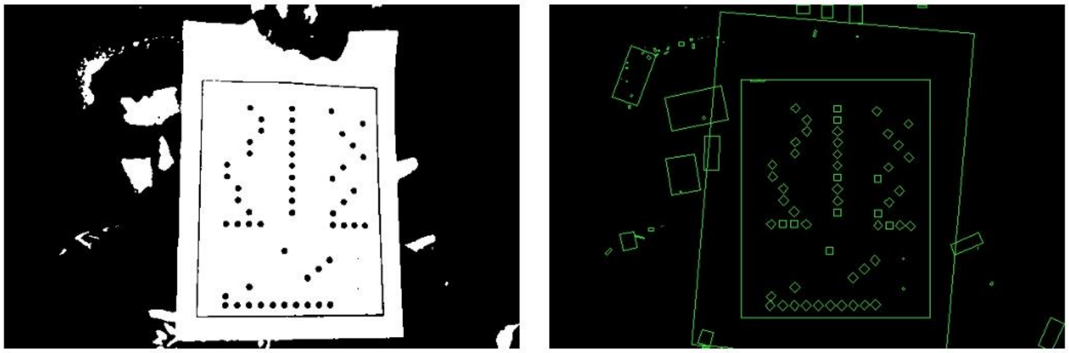


Figure 25: Example of a Rectangle Contour on Page Edges

To cater to the edges detected on the screen boundary, any rectangle whose parameter is along the edges of the screen is considered invalid and removed from the group. And any rectangle which contains another rectangle fully and the inner rectangle is more than 70% of the outer rectangle's area is considered invalid. This eliminates the false largest rectangle and keeps ROI valid.

4.2.2 Cropping Image with Largest Rectangle

The image is cropped with the largest rectangular contours using Affine Transformation. Affine Transformation geometrically transforms the source image to the destination image while preserving lines and parallelism. It takes three input points from the source image and translates them into three points on the destination image. Figure 26 demonstrates how the mapping of three points works.

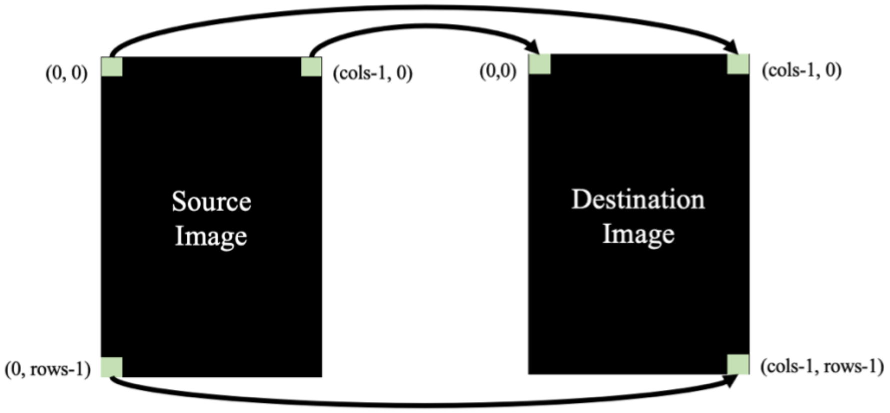
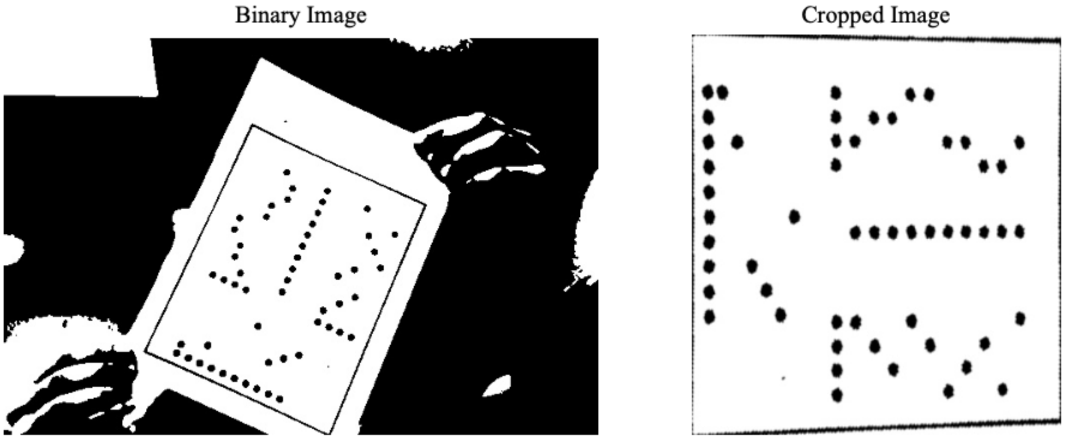


Figure 26: Working of Affine Transformation

While translating the image, the cropped image's orientation will likely be incorrect, as demonstrated in Figure 27. The orientation of the image is corrected in Stage 3 after detecting blobs.


Binary Image

Cropped Image

Figure 27: Example of a Cropped Image with Incorrect Orientation

4.3 Stage 3 – Blob Detection

In the final stage of our methodology, blobs are detected, validated, and compiled into results.

4.3.1 Finding Blobs

The blobs in the cropped-out image, obtained from the previous step, are found using OpenCV's *simpleBlobDetector* function. As explained in Chapter 2, *simpleBlobDetector* detects blobs as connected pixels. Several filtering methods are available that tell the algorithm how to consider the connectivity of pixels, e.g., by area, colour, circularity, etc. For our work, we detected the blobs based on their circularity, which must be filled at least 70% to be considered valid. Figure 28 shows an image with blobs detected via *simpleBlobDetector*, coloured green.

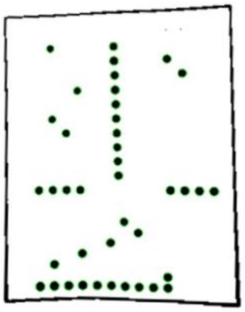


Figure 28: Example of a blob-detected image

4.3.2 Validating Blobs

After blob detection, the next step is to validate if everything until this point has worked okay. Some checks are in place to validate the process, as listed below. If any checks fail, an error message is shown, stating to readjust the image's position. The validation checks are:

- Validating the total number of reference blob points. If the image has four options per question, then a total of 28 reference points must be present.
- Validating the reference blob points for roll number digits. They must be ten equally distant points. It is important to note that the orientation of the image has not been correct yet, so the reference points of the roll number can be on any of the four sides.
- Validating the reference blob points for questions in the centre of the image. There must be ten equally distant points with some error margin.
- Six points for roll number must be present. If the digits for the roll number are less than six, then the leading 0's must be added.
- Above the six points for the student's roll number, there must be eight points along the same x-axis divided into two groups of four points each. These four points must also be equally distant, with some margin of error.
- Total blob points must not exceed the sum of all reference points and the total number of questions (can be a maximum of 20).

4.3.3 Orientation Correction

When blobs are validated, the next step is to correct the orientation of the image. Algorithm 1 summarizes the steps involved in fixing the image's orientation.

Algorithm 1: Orientation Correction of Image

```
$I$ ← input image with incorrect orientation
$C$ ← output image with correct orientation
$bp$ ← blob points detected using simpleBlobDetector
$sbp\_x$ ← sort the $bp$ along $x\_axis$
$ftp\_x$ ← extract first 10 points from $sbp\_x$
if the distance between $ftp\_x$ is equal then
    $C$ ← rotate $I$ by $90°$ in the anti-clockwise direction
    return $C$
end
$ltp\_x$ ← extract last 10 points from $sbp\_x$
if the distance between $ltp\_x$ is equal then
    $C$ ← rotate $I$ by $90°$ in the clockwise direction
    return $C$
end
$sbp\_y$ ← sort the $bp$ along $y\_axis$
$ftp\_y$ ← extract first 10 points from $sbp\_y$
if the distance between $ftp\_y$ is equal then
    $C$ ← rotate $I$ by $180°$
    return $C$
else
    $C$ ← $I$
    return $C$
end
```

4.3.4 Blob Points Rotation

When the image is corrected, the blob points become disarrayed because they contain (x, y) values from the original cropped image. There are two ways to solve this problem. One way is to rerun *simpleBlobDetector* on the correctly oriented image, but it adds to the computational time of the whole system. The other and the faster solution is to rotate each blob point (x, y) along the origin (x_0, y_0) by an angle θ to get a new point (x', y'), defined as:

$$x' = (x - x_0)\, cos\theta - (y - y_0)\, sin\theta + x_0$$
$$y' = (x - x_0)\, sin\theta + (y - y_0)\, cos\theta + y_0$$

4.3.5 Results Compilation

Finally, the markings are found from the detected blob points to compile results. In this phase, firstly, the roll number of the student as (x, y) points is found using Algorithm 2.

Algorithm 2: Finding the student's roll number

```
bp ← detected blob points
r ← digits of roll number to be found
rnrpsi ← starting index of roll number reference points
rnsi ← starting index of roll number digits
for i ← rnsi to rnsi + 6 do
    for j ← rnrpsi to rnrpsi + 10 do
        if $x$ of bp[i] and bp[j] match then
            r = r + bp[i]
            break
        end
    end
end
return r
```

After finding the student's roll number, the marked answers as (x, y) points are detected based on Algorithm 3.

Algorithm 3: Finding the marked answers

```
count ← count of options per each question
o ← 2 × count reference points of options with the same $y$
l ← count options equally distant points from o on the left side
r ← count options equally distant points from o on the right side
c ← 10 equally distant points with the same $y$ in the middle
al ← marked options with $x < c$ and $y < l$
ar ← marked options with $x > c$ and $y < r$
alphabet ← english alphabets (A, B, C, …, count)
answers_dict ← holds key/value pairs of answers

for i ← al start_index to al end_index do
    for j ← c start_index to c end_index do
        if $y$ of al[i] and $y$ of c[j] match then
            for k ← l start_index to l end_index do
                if $x$ of al[i] and $x$ of l[k] match then
                    answers_dict[i] = alphabet
                end
            end
        end
    end
end

answers_dict is updated the same way for questions on the right side
return answers_dict
```

After the *answers_dict* is populated with keys as the question numbers and values as the selected option for each student, it is compared with data from an input file. The input file format is described with some sample data in Table 1.

Data in Input File		***Description***
Line 1	10 (\n)	Number of Questions in Assessment
Line 2	4 (\n)	Number of Options per Assessment. OMR Sheet must be modified if options are less than or more than 4
Line 3	A, C, B, A, A, B, D, D, C, A	Comma-Separated List of the correct answer for each question in ascending order

Table 1: Sample Format of Input File with Correct Answers

The results of the comparison between *answers_dict* and correct answers from the *input file* are pushed into an output .csv file. The format of the output file is described with some sample data in Table 2.

Data in Output File		***Description***
Line 1	Rollno, Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8, Q9, Q10, Total (\n)	The header of the .csv file
Line 2	144048, A, B, B, A, A, C, D, D, B, A, 7 (\n)	First Student's Answer Sheet
Line 3	144225, A, C, B, A, A, B, D, D, C, A, 10	Last Student's Answer Sheet

Table 2: Sample Format of Output File with Markings

Chapter 5 – Experiments and Results

This chapter discusses the information about our data collection, our experiments, and the results obtained from those experiments. We conducted experiments with different parameters to check which scenarios work best with our proposed strategy and which fail.

5.1 Environmental Setup

Table 3 states the main parameters we set up for our experiments.

Webcam Type	Built-in webcam on MacBook Pro 2015
Webcam Resolution	720p (1280x720 pixels or 0.9 megapixels)
Webcam Aspect Ratio	16:10
Room	Indoor (with and without incoming outdoor light)
Distance from webcam	Between 24 to 36 inches
Light Source	Warm White, White, and Cool White Bulbs Colour Temperatures ranging from 2700K to 4000K
Light Illumination	Ranging between 25 lux to 1400 lux

Table 3: Environmental Setup

The environmental setup for light illumination was carried out using a digital light meter [24], whose range was from 0 to 200,000 lux.

5.2 Dataset

For the dataset, we assembled a set of 1280 images in total. The composition of this dataset is explained in Table 4.

20	Unique Solutions	
8	Orientation Angles	Portrait 0°, Landscape 90°, Reverse Portrait 180°, Reverse Landscape 270°, Between $0^\circ - 90^\circ, 90^\circ - 180^\circ, 180^\circ - 270^\circ, 270^\circ - 360^\circ$
2	Skewness Angles	Straight 0°, Between $0^\circ - 45^\circ$
4	Lighting Conditions	Warm White (25 – 150 lux) White (25 – 250 lux) Cool White (50 – 300 lux) Room with daylight (150 – 1400 lux)
$20 \times 8 \times 2 \times 4 = 1280$		

Table 4: Dataset Composition

5.3 Evaluation Criteria

The proposed system is evaluated based on two criteria:

- Effectiveness of image binarization of proposed algorithm and other binarization algorithms.
- Comparison between the proposed technique and other existing methods.

5.3.1 Effectiveness of Proposed Binarization Algorithm

Image Binarization between the proposed algorithm and other algorithms can be visually judged based on noise. If the resultant binary image has lesser noise and less unrequired information than the others, it performs well.

However, it can be objectively compared based on the number of warning messages generated on each image in the dataset. If a binarization algorithm gives a minimum number of warning messages on the dataset, it is better than other algorithms.

5.3.2 Effectiveness of Complete Proposed System

The criteria to evaluate the effectiveness of the complete proposed system can be divided into two categories: the accuracy of an individual sheet, the accuracy of the whole system

For an Individual Sheet:

The confusion matrix for finding the accuracy of an individual sheet is given in Table 5.

True Positive	The system identified the correct option for its respective question
False Positive	The system identified an incorrect option for its respective question
True Negative	The system identified the correct left-out option for its respective question
False Negative	The system identified an incorrect left-out option for its respective question

Table 5: Confusion Matrix for finding the accuracy of an Individual Sheet

$$Accuracy_{Ind} = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

For the whole system:

If the total number of sheets tested with the system is represented as I_{Total}, and the number of sheets for which the system generated correct output, i.e., their $Accuracy_{Ind}$ is 100, are represented as $I_{Correct}$, and the number of sheets for which the system prompted a warning message is represented as $I_{Warning}$, then the accuracy of the complete system is defined as:

$$Accuracy_{System} = \frac{I_{Correct}}{I_{Total} - I_{Warning}} \times 100$$

Acceptance Criteria:

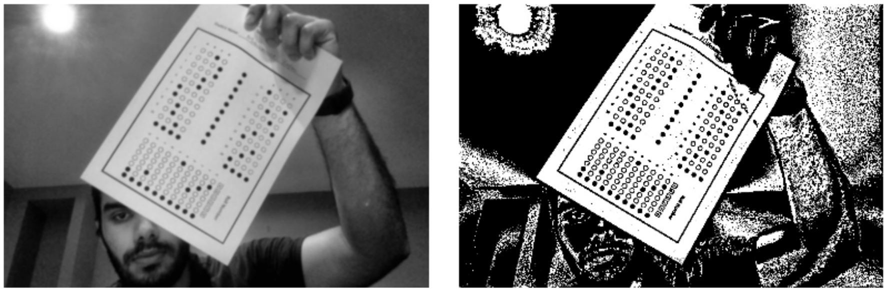
OMR in education is a sensitive application because its users cannot afford to get incorrect results. Therefore, $Accuracy_{System}$ must be 100% for the proposed system to be considered valid.

5.4 Results

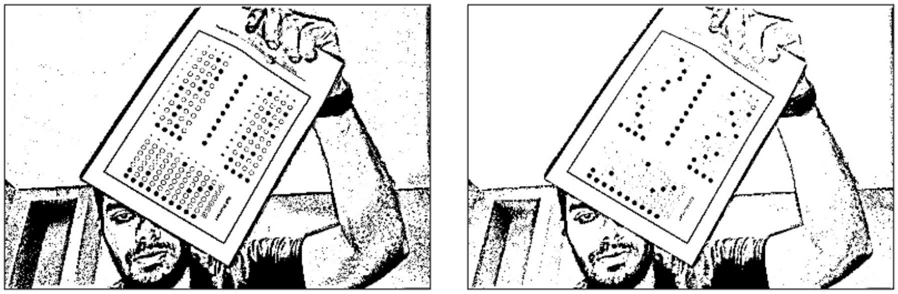
As a result, we will first analyze the performance of our proposed binarization algorithm and other algorithms.

5.4.1 Proposed Binarization Algorithm Test

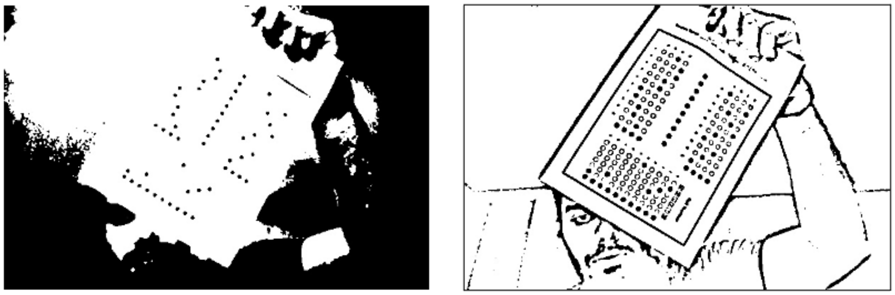
Figure 29 and Figure 30 show examples of the comparison between our proposed binarization algorithm and several different algorithms. In figure 29, our proposed Otsu variant averages the value between the original Otsu and Otsu with the neighbourhood. In contrast, in figure 30, our Otsu gives the same threshold value as the original Otsu.


Grayscale Image

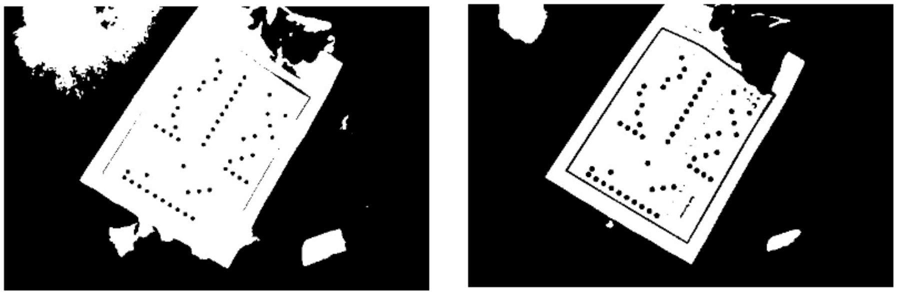
Niblack (window = 30x30, k = -0.2)


Sauvola (window = 30x30, k = 0.5)

Wolf & Jolion (window = 30x30, k = 0.5)


Simple (t: 128)

Adaptive (BlockSize: 25, Constant: 10)


Original Otsu (t = 134)

Otsu with neighbourhood (t = 184)

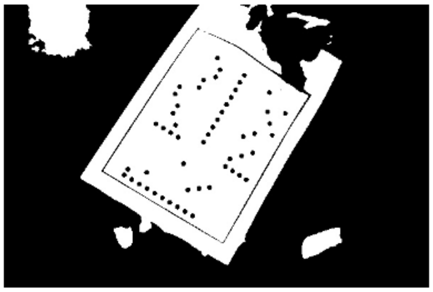
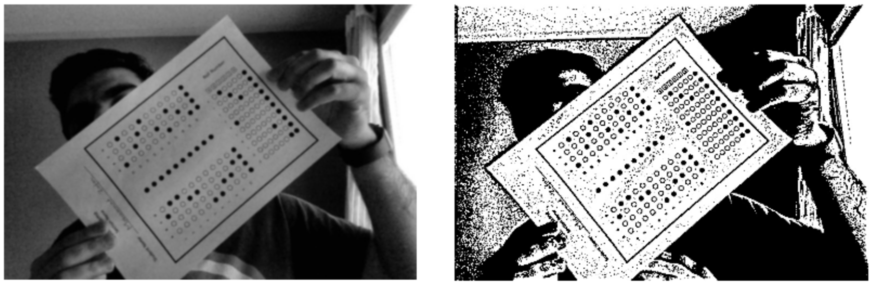

Proposed Otsu Binarization (t = 159)

Figure 29: Example 1 of different binarization algorithms

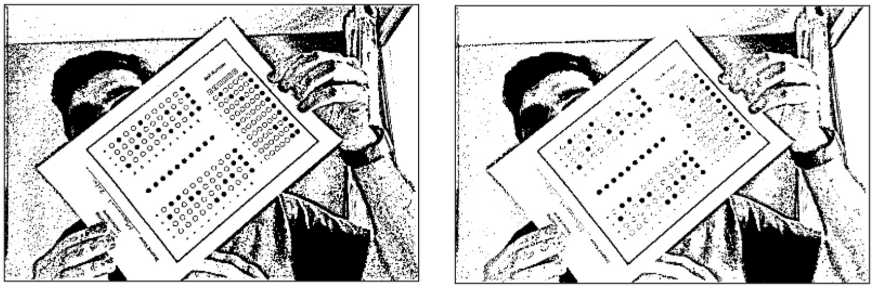
Grayscale Image

Niblack (window = 30x30, k = -0.2)



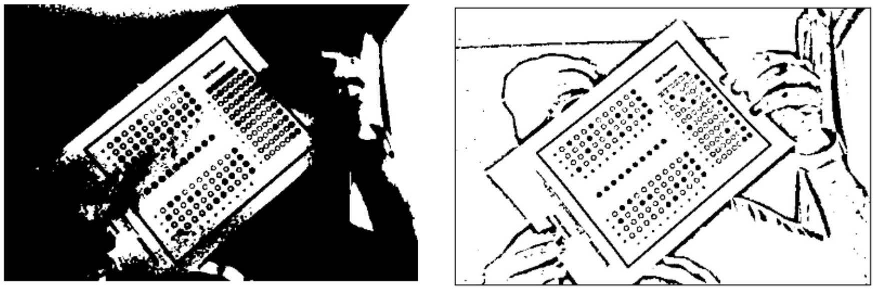
Sauvola (window = 30x30, k = 0.5)

Wolf & Jolion (window = 30x30, k = 0.5)



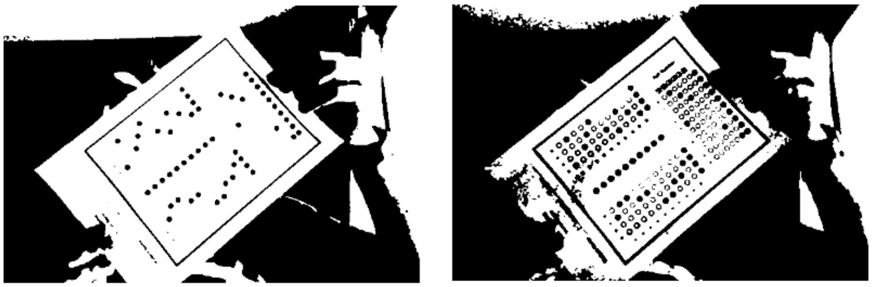
Simple (t: 128)

Adaptive (BlockSize: 25, Constant: 10)



Original Otsu (t = 87)

Otsu with neighbourhood (t = 120)



Proposed Otsu Binarization (t = 87)

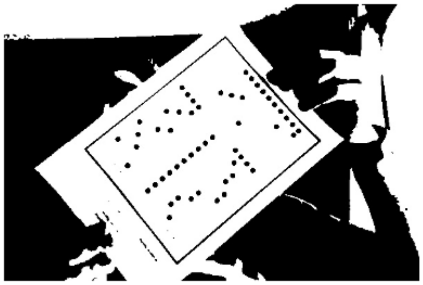


Figure 30: Example 2 of different binarization algorithms

200 images were taken from our dataset of 1280, with random illumination and random angles or rotation. Figure 31 depicts a chart showing the performance of each algorithm on that 200-image dataset. It shows the percentage of warning messages prompted by each algorithm—the lesser the percentage, the better the binarization. Table 6 shows the pre-defined values which were used by some of the algorithms.

Simple Binarization	T = 128
Adaptive Binarization	Blocksize = 25, Constant = 10
Niblack Binarization	Window = 30x30, k = -0.2
Sauvola Binarization	Window = 30x30, k = 0.5
Wolf & Jolion Binarization	Window = 30x30, k = 0.5

Table 6: Pre-defined Value used for Binarization Algorithms

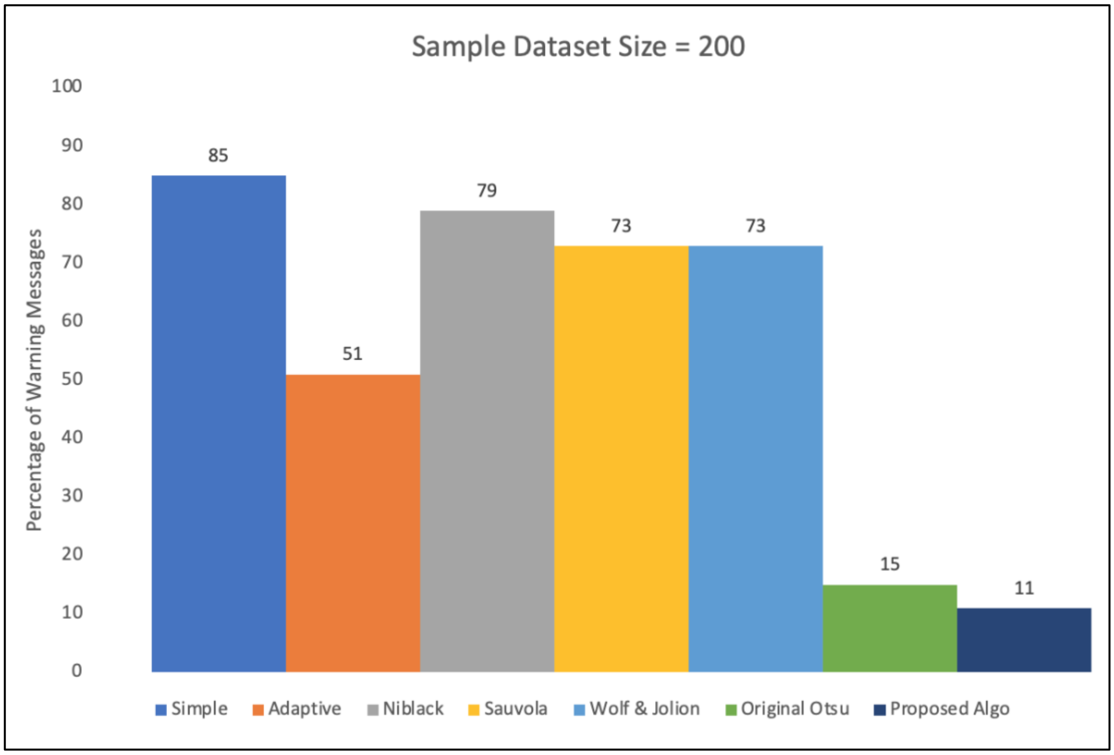


Figure 31: Chart showing a comparison between binarization algorithms

Although by changing some pre-defined values defined in Table 6, the results improved a bit. But the improvement was ignorable, as no specific value could have generated results close to an automated binarization algorithm like Otsu Binarization or our proposed algorithm.

Next, we plotted a chart stating the time taken by each of these binarization algorithms. Unsurprisingly, simple binarization took the least time because it has the least amount of pixel manipulations. But the Otsu Binarization and our proposed algorithm were also not very computationally expensive. Figure 32 shows the plot of algorithms with their time taken.

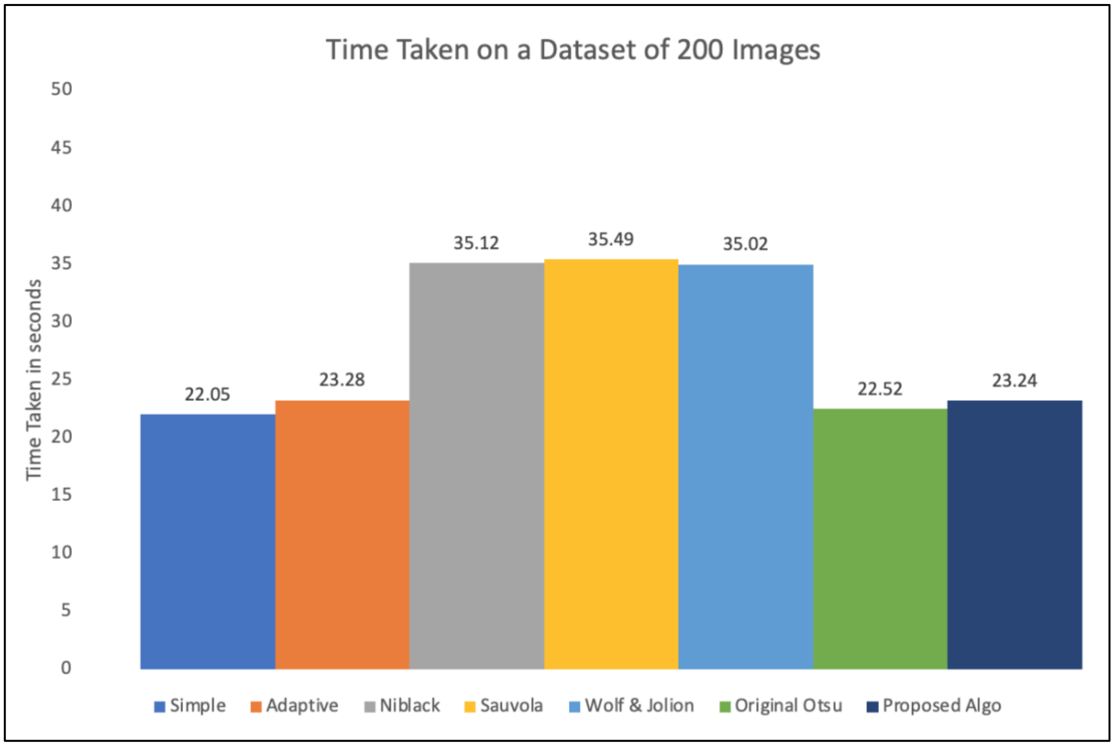


Figure 32: Chart showing the time taken by different binarization algorithms

Based on our findings with these algorithms, Niblack, Sauvola, and Wolf & Jolion Binarization Algorithms take more time than the rest, and their parameters must be adjusted to get good results. Simple Binarization and Adaptive Binarization had a reasonable amount of computational time, but the parameters must be adjusted to yield good results. The results from Otsu Binarization and our proposed algorithm were close. But in the case of dark or bright images, our proposed algorithm has a slight advantage because of considering the pixel-neighbourhood.

5.4.2 Proposed System Complete Test

After investigating the results of our thresholding algorithm, the next step is to analyze the performance of the complete system end-to-end. First, we will examine our system's performance with different parameters and settings. Then, we will compare our proposed model with other models.

5.4.2.1 Test of Basic Functionality:

In Table 7, we have 8 images with 2 unique solutions. Each solution is captured in 4 orientations, i.e., Portrait, Landscape, Reverse Portrait, and Reverse Landscape. Both images have been held straight toward the camera with no skewness. And the illumination for the first image is between 50 – 70 lux, while for the second image, it is between 30 – 40 lux.

Image	***Orientation***	***Correct***	***Correctly Detected***	$\boldsymbol{Accuracy_{Ind}}$
1	Portrait	10	10	100
2	Landscape	10	10	100
3	Reverse Portrait	10	10	100
4	Reverse Landscape	10	10	100
5	Portrait	7	7	100
6	Landscape	7	7	100
7	Reverse Portrait	7	7	100
8	Reverse Landscape	7	7	100
$Accuracy_{System} = 100$				

Table 7: Results to test basic functionality

5.4.2.2 Test of Orientation:

In Table 8, we combined analysis from multiple sets of images with the same orientation inside each group but different across multiple groups. Each group contains 20 images, all unique solutions. All images were straight with no skewness. And the illuminance of all images inside all sets varies between 25 to 250 lux.

In the table, *Correct* represents the number of sheets per set correctly detected by the OMR system, i.e., $Accuracy_{Ind}$ for the sheet is 100. *Warning* represents the number of sheets for which the system has generated a warning message.

Image Set	***Orientation***	***Correct***	***Warning***	$\boldsymbol{Accuracy_{System}}$
1	Portrait	20	0	100
2	Landscape	20	0	100
3	Reverse Portrait	20	0	100
4	Reverse Landscape	20	0	100
5	Between $0° - 90°$	20	0	100
6	Between $90° - 180°$	20	0	100
7	Between $180° - 270°$	20	0	100
8	Between $270° - 360°$	20	0	100

Table 8: Results to test orientation

5.4.2.3 Test of Skewness:

In Table 9, we have 8 unique images with the same orientation (portrait). Each image is held at a different angle to assess its effect on performance. And the illuminance of all images varies between 25 to 250 lux.

Correct represents the number of questions that the student correctly marked, *Correctly Detected* represents the number of questions that were accurately detected by the system, and *Warning* represents whether the system has generated the warning message.

Image	***Skewness***	***Correct***	***Correctly Detected***	***Warning***	$\boldsymbol{Accuracy_{Ind}}$
1	Between $0° - 5°$	10	10	-	100
2	Between $5° - 10°$	8	8	-	100
3	Between $10° - 15°$	5	5	-	100
4	Between $15° - 20°$	10	10	-	100
5	Between $25° - 30°$	8	8	-	100
6	Between $30° - 35°$	6	0	Yes	100
7	Between $35° - 40°$	10	0	Yes	100
8	Between $40° - 45°$	8	0	Yes	100
$Accuracy_{System} = 100$					

Table 9: Results to test skewness

5.4.2.4 Test of Skewness and Orientation (combined):

In Table 10, we took 8 unique images with different orientations and skewness. And the illuminance of all images varies between 25 to 250 lux.

Image	***Orientation***	***Skewness***	***Correct***	***Correctly Detected***	***Warning***
1	Portrait	Between $15° - 30°$	10	10	-
2	Landscape	Between $0° - 15°$	10	10	-
3	Reverse Portrait	Between $30° - 45°$	7	0	Yes
4	Reverse Landscape	Between $30° - 45°$	9	9	-
5	Between $0° - 90°$	Between $15° - 30°$	1	1	-
6	Between $90° - 180°$	Between $30° - 45°$	5	0	Yes
7	Between $180° - 270°$	Between $30° - 45°$	6	0	Yes
8	Between $270° - 360°$	Between $0° - 15°$	9	9	-
$Accuracy_{System} = 100$					

Table 10: Results to test the combination of skewness and orientation

5.4.2.5 Test of Lighting Conditions:

In Table 11, we took 9 unique images with the same orientation (portrait) and same skewness (straight) to test the effect of lighting conditions on our results. In *Environment,* there is a mention of whether the sheet was facing the light in front or back.

Image	***Lux***	***Environment***	***Correct***	***Correctly Detected***	***Warning***
1	25 – 50	Warm Light (back)	5	5	-
2	110 – 120	Warm Light (front)	9	9	-
3	150 – 170	White Light (front)	4	4	-
4	220 – 230	Cool Light (front)	10	10	-
5	180 – 200	Daylight (overcast – back)	8	8	-
6	650 – 670	Daylight (overcast – front)	9	9	-
7	800 – 850	Daylight (sunny – back)	10	10	-
8	1400 – 1420	Daylight (sunny – front)	10	10	-
9	38000	Sunlight (front)	10	0	Yes
10	1 – 10	Very Dim Light	10	0	Yes
$Accuracy_{System} = 100$					

Table 11: Results to test lighting conditions

5.4.2.6 Test of Lighting Conditions and Orientation (combined):

In Table 12, we have 10 unique images with different orientations but the same skewness (straight) to test the effect of lighting conditions on our results.

Image	***Lux***	***Orientation***	***Correct***	***Correctly Detected***	***Warning***
1	25 – 100	Portrait	8	8	-
2	50 – 150	Landscape	9	9	-
3	250 – 300	Reverse Portrait	3	3	-
4	1400 – 1450	Reverse Landscape	9	9	-
5	300 – 350	Between $0° - 90°$	10	10	-
6	800 – 900	Between $90° - 180°$	10	10	-
7	100 – 200	Between $180° - 270°$	2	2	-
8	1000 – 1100	Between $270° - 360°$	4	4	-
9	400 – 450	Reverse Portrait	10	10	-
10	1300 – 1350	Reverse Landscape	10	10	-
$Accuracy_{System} = 100$					

Table 12: Results to test the combination of lighting conditions and orientation

5.4.2.7 Test of Lighting Conditions and Skewness (combined):

In Table 13, we have 8 unique images with the same orientation (portrait) but different skewness to examine the effect of lighting conditions on our results.

Image	*Lux*	*Skewness*	*Correct*	*Correctly Detected*	*Warning*
1	25 – 40	Between 0° – 15°	3	3	-
2	25 – 40	Between 30° – 45°	8	0	Yes
3	50 – 100	Between 15° – 30°	1	1	-
4	150 – 180	Between 0° – 15°	10	10	-
5	200 – 250	Between 0° – 15°	4	4	-
6	130 – 160	Between 30° – 45°	5	0	Yes
7	80 – 100	Between 30° – 45°	7	7	-
8	400 – 450	Between 0° – 15°	1	1	-
9	800 – 830	Between 30° – 45°	10	0	Yes
10	800 – 830	Between 15° – 30°	8	8	-
$Accuracy_{System} = 100$					

Table 13: Results to test the combination of lighting conditions and skewness

5.4.2.8 Test on Complete Dataset

Finally, our proposed system is tested on the complete dataset of 1280 images to check the overall accuracy. Table 14 shows the result of running our proposed system with the dataset.

Total	***Correct***	***Warning***	***Incorrect***	***Average Time Per Image (sec)***	$Accuracy_{System}$
1280	1106	174	0	0.18	100

Table 14: Results to test the complete dataset

In comparison, we also inspected the performance of other binarization algorithms on our dataset, as shown in Table 15. As we have validations in place after the image is binarized, none of the following algorithms will give incorrect results. Their performance can only be judged on the number of warnings.

Algorithm	***Correct***	***Warning***	***Incorrect***	***Average Time Per Image (sec)***
Otsu	1029	251	0	0.17
Simple	167	1113	0	0.17
Adaptive	653	627	0	0.18
Niblack	265	1015	0	0.24
Sauvola	316	964	0	0.24
Wolf & Jolion	324	956	0	0.23

Table 15: Results of other binarization algorithms

5.4.3 Comparison of the Proposed System with Existing Systems

Finally, we compare the performance of our system, based on different parameters, with existing systems. Table 16 summarizes the differences among all. In the table, *Camera Resolution* represents the camera's resolution in megapixels to keep things consistent with other research works. So, a 720p becomes $(1280 \times 720/1000000) = 0.9$mp.

Method	***Number of Images***	***Accuracy***	***Time taken per sheet (seconds)***	***Camera Type & Resolution***	***Lighting***
Talib et al. [17] (2015)	5	97.5	N/A	Webcam (1.3)	Good Light Required
Karunanayake [18] (2015)	3	97.6	2	Webcam (N/A)	Good Light Required
Patel et al. [8] (2015)	310	98.2	3	Mobile (N/A)	300 – 400
Rasiq et al. [19] (2019)	5	99.44	N/A	Mobile (N/A)	300 – 400
Jingyi et al. [21] (2021)	3	97.72	N/A	Mobile (N/A)	Good Light Required
Raveendran [22] (2022)	100	100	0.6	Webcam (2)	400 – 800
Proposed System	1280	100	0.18	Webcam (0.9)	25 – 1400

Table 16: Comparison between the proposed system and existing methods

5.5 Observations from Experiments

Based on our experiments and the results produced, we have made the following observations:

- Orientation of the sheet while capturing its image does not affect the system's performance.
- Lighting Conditions have an effect when very dark, making it difficult for the algorithm to distinguish the foreground from the background. Also, it faces difficulties when the light is too bright. When the light is bright, the resulting image glares, making the thresholding algorithm lose some detail about the markings. Thus, yielding warning messages.
- The angle of the sheet while capturing its image has the most effect on the system. The system performs well if the angle is contained between $0^\circ\ to\ 30^\circ$. But when the angle exceeds 30°, the number of warning messages also exceeds.

Summary:

The sheet can be captured in any orientation, but the skewness must be within a constraint of $0^\circ\ to\ 30^\circ$ to minimize the number of warning messages. And the lighting conditions can vary between 25 lux to 1400. This range of illuminance levels covers a room with dull lighting (not very dark) to a room with daylight (sunny day) coming in.

Chapter 6 – Conclusion and Future Work

Optical Mark Recognition has been studied for almost a century. This is because of the need for this technology in many sectors to save time and money. Over the years, OMR has shifted from hardware-based to software-based systems, generating many opportunities for research in this area.

Our primary goal was to design a low-cost webcam-based OMR system to help instructors save time and money worldwide. We proposed a variant of the Otsu Binarization Algorithm that can work well in a dynamic environment and generate results with 100% accuracy. We compared our system with other existing systems to validate its performance. We tweaked different parameters, such as the orientation of the OMR sheet, its angle, and lighting changes, to examine our system's resilience.

This system can be extended for future work to video, where the application is running. The instructor just swaps different OMR sheets in front of their camera without clicking any other button. The application will determine which frame to capture and when the OMR sheet has been changed.

Also, the format of the OMR sheet can be generated automatically. Instead of creating a manual template, the user must be able to dictate how many questions they need, the options per question, and the digits of students' roll numbers. The system should be able to generate an OMR sheet based on user input.

Bibliography

[1] Zhang, L., Li, B., Zhang, Q., & Hsiao, I.-H. (2020). Does a Distributed Practice Strategy for Multiple Choice Questions Help Novices Learn Programming? *International Journal of Emerging Technologies in Learning (IJET), 15*(18), 234-250. https://doi.org/10.3991/ijet.v15i18.10567

[2] *Press release*. ITU. (2021, November 29). ITU. https://www.itu.int/en/mediacentre/Pages/PR-2021-11-29-FactsFigures.aspx. Last accessed January 18, 2023.

[3] Smith, A. M. (1981). Optical mark reading - making it easy for users. *Proceedings of the 9th Annual ACM SIGUCCS Conference on User Services - SIGUCCS '81,* 257-263. https://doi.org/10.1145/800079.802600

[4] *IBM Archives: IBM 805 Test Scoring Machine*. https://www.ibm.com/ibm/history/exhibits/specialprod1/specialprod1_9.html. Last accessed January 18, 2023.

[5] *A Bubble for Your Thoughts? Not for Long*. (2013, August 4). Timeline – NYTimes.com. https://archive.nytimes.com/www.nytimes.com/interactive/2013/08/04/education/edlife/20130804_edlife_testing.html. Last accessed January 18, 2023.

[6] Veronese, K. (2015, December 16). *The birth of Scantrons, the bane of standardized testing*. Gizmodo. https://gizmodo.com/the-birth-of-scantrons-the-bane-of-standardized-testin-5908833. Last accessed January 18, 2023.

[7] Gravic Inc. (2007). *Remark Office Omr: User's Guide*.

[8] Patel, R., Sanghavi, S., Gupta, D., & Raval, M. S. (2015). CheckIt - a low cost mobile OMR System. *TENCON 2015 - 2015 IEEE Region 10 Conference*, 1-5. https://doi.org/10.1109/tencon.2015.7372983

[9] Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(1), 62-66. https://doi.org/10.1109/tsmc.1979.4310076

[10] Niblack, W. (1988). *An introduction to digital image processing*. Prentice Hall International.

[11] Sauvola, J., & Pietikäinen, M. (2000). Adaptive document image binarization. *Pattern Recognition*, *33*(2), 225-236. https://doi.org/10.1016/s0031-3203(99)00055-2

[12] Wolf, C., Jolion, J. M., & Chassaing, F. (2002). Text localization, enhancement and binarization in multimedia documents. *Object Recognition Supported by User Interaction for Service Robots*, vol. 2, 1037-1040. https://doi.org/10.1109/icpr.2002.1048482

[13] Bradski, G., & Kaehler, A. (2012). *Learning opencv: Computer vision with the opencv library*. O'Reilly.

[14] *OpenCV: cv::SimpleBlobDetector Class Reference*. https://docs.opencv.org/3.4/d0/d7a/classcv_1_1SimpleBlobDetector.html. Last accessed January 18, 2023.

[15] *OpenCV: Contours: Getting Started*. https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html. Last accessed January 18, 2023.

[16] *OpenCV: Contour Features.* https://docs.opencv.org/3.4/dd/d49/tutorial_py_contour_features.html. Last accessed January 18, 2023.

[17] Talib, A., Ahmad, N., & Tahar, W. (2015). OMR Form Inspection by Web Camera using Shape-Based Matching Approach. *International Journal of Research in Engineering and Science, 3*(4), 29-35.

[18] Karunanayake, N. (2015). OMR Sheet Evaluation by Web Camera Using Template Matching Approach. *International Journal for Research in Emerging Science and Technology*, *2*(8), 40-44.

[19] Rasiq, G. R. I., Al Sefat, A., & Hasnain, M. F. (2019). Mobile based MCQ Answer Sheet Analysis and Evaluation Application. *2019 8th International Conference System Modeling and Advancement in Research Trends (SMART)*, 144-147. https://doi.org/10.1109/smart46866.2019.9117468

[20] Afifi, M., & Hussain, K. F. (2019). The achievement of higher flexibility in multiple-choice-based tests using image classification techniques. *International Journal on Document Analysis and Recognition (IJDAR)*, *22*(2), 127–142. https://doi.org/10.1007/s10032-019-00322-3

[21] Jingyi, T., Hooi, Y. K., & Bin, O. K. (2021). Image processing for enhanced OMR answer matching precision. *2021 International Conference on Computer & Information Sciences (ICCOINS)*, 322-327. https://doi.org/10.1109/iccoins49721.2021.9497172

[22] Raveendran, R. B. (2022). *Effective auto grading with webcam* (Order No. 29214256). Available from Dissertations & Theses @ University of Windsor; ProQuest Dissertations & Theses Global. (2682012430).

[23] Quan, Y., Sun, J., Zhang, Y., & Zhang, H. (2019). The Method of the Road Surface Crack Detection by the Improved Otsu Threshold. *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, 1615-1620. https://doi.org/10.1109/icma.2019.8816422

[24] *Digital Light Meter, LX1330B,* Dr.meter. https://drmeter.com/products/lx1330b-digital-illuminance-light-meter. Last accessed January 18, 2023.

Vita Auctoris

NAME:	Muhammad Fahad Zafar
PLACE OF BIRTH:	Khanewal, Pakistan
YEAR OF BIRTH:	1996
EDUCATION:	Bachelor of Science in Computer Science National University of Computer and Emerging Sciences Lahore, Pakistan, 2014 – 2018
	Master of Science in Computer Science Co-op University of Windsor Windsor, ON, Canada, 2021 – 2023