OMR Sheet Evaluation Using Image Processing

Prof. Pramod Kanjalkar, Prasad Chinchole, Archit Chitre, Apoorv Damale, Harsh Chandak

Department of Instrumentation and Control Engineering

Abstract — Optical Mark Recognition (OMR) Sheet, also called as Bubble sheet is a special type of form used to answer graded multiple choice question examinations, where students have to mark or darken the bubbles to answer the questions. OMR sheets are used by various school, college, university and competitive examinations. Larger institutions like universities have specialized machines to optically detect the marked answers and grade the student. However, these scanners are significantly heavy on the pocket and cannot be afforded by small tuitions or individual teachers. This paper presents a cost-effective solution to accurately scan OMR sheets without the need of scanners.

Keywords — OMR, Image processing, OpenCV, Numpy, Python, Web application, Flask

I. Introduction

The automatic technique of capturing data in the form of marks such as bubbles or ticks is known as Optical mark Recognition. OMR based evaluation is preferred over manual methods when a large volume of data is to be collected. OMR widely finds its applications in graded multiple-choice examinations where students are required to mark their answers by darkening the circles on a pre-printed paper. The sheet is then evaluated using image scanning machine. These machines shine a beam of light onto the paper and takes into account the difference in the density of light reflected back as the marked region reflects less light than the unmarked regions. However, they cannot detect the shape of the mark.

However, these machines cost anywhere between Rs. 25,000 – Rs. 30,000 and can only be afforded by bigger institutions like schools or universities. Although popular examinations conducted in India like the *JEE* or *NEET* have moved onto online solutions and conduct examinations on their own online portal.

In case of small classes, tuition groups or individual teachers, with limited economic capacity, buying these scanners or setting up their own online portals would generally not be an option. Even in various developing countries, buying and maintaining these devices becomes increasingly expensive. Therefore, this paper presents a full-fledged and cost-effective solution to grade students quickly and efficiently using Image Processing techniques. This paper presents an image processing algorithm to scan an answer sheet and detect the responses given by a student. The work presented in this paper is implemented in Python for the image processing using the OpenCV library. This entire project is deployed as a web application for the end user to use. The frontend is developed using HTML, CSS and JavaScript, the Flask framework as the backend and SQLite as the database. The user is expected to scan all the answer sheets, upload them to the portal using their laptop/mobile and automatically see the results of all scanned sheets by comparing the answer key stored in the database. This method is very efficient in case of a large number of students where evaluating manually would be tedious task.

The main aim of the project presented in this paper is to use various image processing techniques to automate the entire process of evaluating an OMR sheet quickly and efficiently. This project is deployed on Heroku, a popular cloud-based Platform-as-a-Service. This project can be found on the following link:

II. Literature Review

The method used by Ismael Amara Belag, Yasemin Gultepe and Tark Melude Elmalti is based on creation of template answer sheet and key points detection algorithm. The proposed method automatically computes the number of correct answers using vertical and horizontal projections and thresholds techniques. The proposed method is able to detect more than one or no selected choice. In this study they have tested more than 100 exam papers. [1]

This paper proposed OMR based on Modify Multi-Connect Architecture (MMCA) associative memory, its work in two phases; training and recognition phase. The proposed method was also able to detect more than one or no selected choice. Among 800 test samples with 8 types of grid answer sheets and total 58000 questions, the system exhibits an accuracy is 99.96% in the recognition of marked.[2]

In this study, discussed method is called the recognition of optical marks and is the process of capturing the data on the multiple-choice forms. The application of recognition has been developed by using the software language, Python and the image processing library, OpenCV. Also, Imutils and ZBar libraries used. An application that prints out the information that is in a QR Code on the optical form in a file together with the answers

on the optical form marked by the student has also been developed. [3]

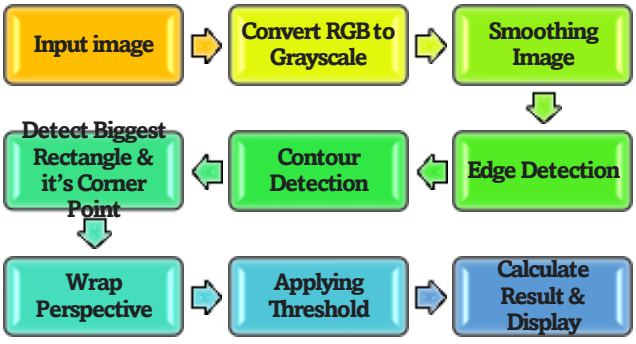
A survey by Nirali Patel and Ghanshyam Prajapati says that the OMR (with simple scanner) have extensive use in small organization as well as in big organizations. While in multi-core processors 4 different types of classifiers namely, Bayes classifier, QDF, MQDF, and NN is used for reduced the processing times of different phases due to parallelization. By using OpenCV libraries and webcam one can discover all the answers and solve the OMR Sheet calculate area of the mark and have confidence level of point nine which will cater accurate judgments.[4]

Here, Nalan Karunanayake has proposed a cost-effective OMR sheet evaluation system based on a low-priced web camera that can evaluate any format of MCQ paper efficiently. First, the selected region with all correct answers is marked on the printed paper separated manually and will be used as a template image in the matching process to extract the answered region of the student answer script. Then the cropped region of the answer sheet is matched with the template image to recognize the answers marked as correct or incorrect. Author has tested with three different formats of MCQ papers, basically containing different number of answers in a column. The results obtained had accuracy of 97.6%. Also, the author noted that bubbles marked with pen give higher accuracy than those marked with pencil.[5]

Apart from the above research, Pune based Nikita Kakade implemented OMR sheet evaluation using image processing but the exception is that she also designed a complimentary hardware setup for this research mainly consisting of conveyer belt and microcontroller. OMR answer sheet will be placed on the conveyor belt which will move on the command from the microcontroller. Once the sheet comes under the webcam, a snapshot is taken. This snapshot is treated as an input image. Then image processing techniques are implemented. This is compared with the database and results are compared. [6]

III. METHODOLOGY

The steps in the methodology of the research proposal are depicted in the flow chart below.



1. Convert RGB Image to Gray Scale

Input images are images of students' response sheets. Since OMR answer sheets come in a variety of colors, it's required to transform them to a uniform color for the rest of the process to run properly. As a result, we use the RGB to grey scale image to get the grey image of the input image.

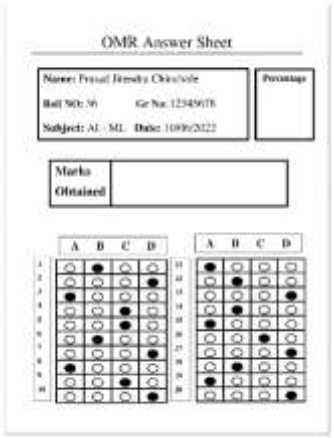


Fig 1. Gray Scale Image

2. Image Smoothing

The image is blurred by using a low-pass filter kernel to convolve it. It can be used to reduce noise. The Gaussian filter is used to lower the high-frequency components. When it comes to removing noise from an image, Gaussian blurring is extremely effective technique.

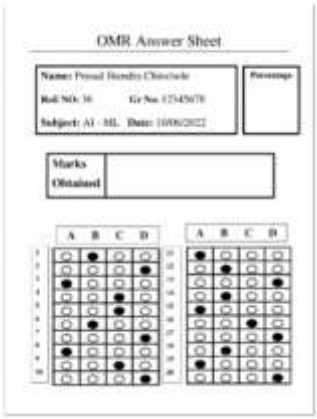


Fig 2. Blur Image

3. Detecting Edges

Edge detection is a type of image processing that identifies the edges of objects or regions inside a picture. Edge detection necessitates computing numerical derivatives of pixel intensities, which generally results in 'noisy' edges. To put it in another way, the intensity of neighboring pixels in an image can vary a lot, resulting in edges that don't represent the dominating edge structure we're searching for.

Blurring smoothens the intensity variations around the edges, making it simpler to recognize the image's dominant edge structure. For detecting the edges in the image, we use Canny Edge Detection. Because of its robustness and flexibility, it is one of the most often used edge detection algorithms.

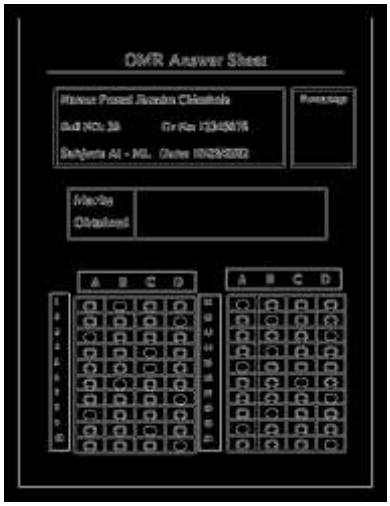


Fig 3. Canny Edge Detection

4. Finding Rectangular Contours

Using contour detection, we can easily identify an object's outlines in a photograph. To discover the contours in a picture, use the findContours() function. Use the drawContours() function to place the contours on the input RGB image once they've been found.

The contourArea() function is used to sort distinct rectangles by their area. Then, from the available rectangles, we pick the biggest rectangles. Because we require the external conner points of rectangles, we use cv2.RETR EXTERNAL. Then we reorder the corner points.

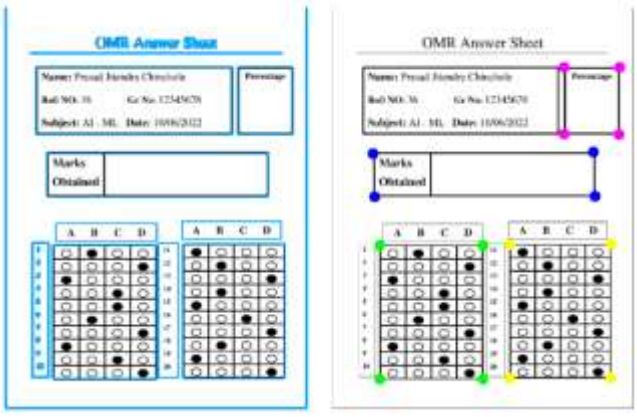


Fig 4. Biggest Contours

5. Warp Perspective of the Bubbled Area

The transformation or perspective correction of images from angle to bird's eye view transform is known as perspective view warping. We used the warpperspective() function to extract the required rectangular portion to a bird's eye view from the Input Image.

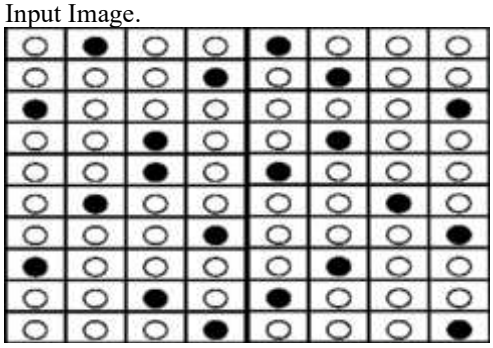


Fig 5. Warp Perspective Of Marked Area

IV. Results and Discussions

Once we have the indices which are marked by the student, it is compared to the actual answer key fed by the teacher. The result is displayed on the actual image, by coloring the correctly marked bubbles in green, wrongly marked answers by red and the actual answer in yellow

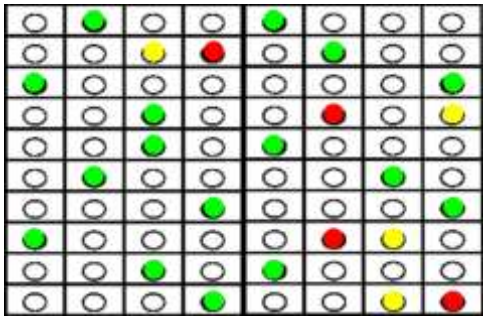


Fig 6. Correct answers are marked green, wrong answers by red and actual answers by yellow on the original image.

Finally, the calculated marks and percentage are displayed on the original image. The following figure is the final image which will be sent as the response back to the browser by encoding the image into a bytestring so that it can be readable by the 'src' attribute of the HTML image tags.

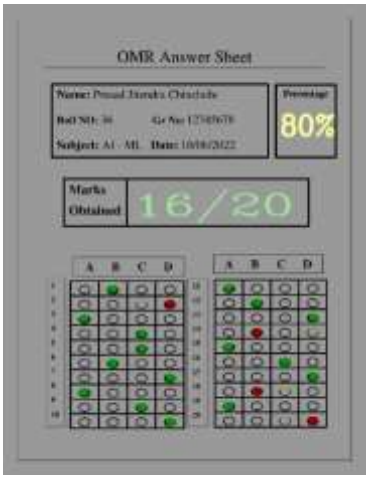


Fig 7. The final image with the marks and percentage displayed

V. Web Application

A web application is developed for the end user to use this system. The user is expected to send all the images at once, the images will be sent to the server where the algorithm resides and an appropriate response will be sent back.

The frontend is developed using HTML, CSS and JavaScript. Bootstrap, an open-source CSS framework was also used for many components to create a responsive and mobile-friendly web application. For the backend, Flask, a small and light-weight Python based framework is used. Flask provides a variety of useful tools and features to make web applications much faster. As for the database, SQLite3 is used. SQLite belongs to the family of embedded databases. It is a library that one can embed in their apps. The main advantage of SQLite is that is does not require a server to fetch the data. The entire database is stored locally on the machine.

The server side code is written in the *app.py,* where the server is created and configured. The image processing algorithm is contained in the file *main.py* which is imported in the app.py file. Various endpoints have been defined and each endpoint when requested sends an appropriate response back to the user. All scanned images are sent to the "/upload-answer-sheets" route where each image is fed into the algorithm and the data (marks) is returned.

The database contains one table – Test which has 5 fields:

1. sno (INTEGER) – Serial number which also serves as the Primary Key of the table.
2. name (STRING) – Test name entered by the teacher
3. answer_key1 (STRING) – Answer key of the first 10 questions
4. answer_key2 (STRING) – Answer key of the next 10 questions
5. datetime (DATETIME) – Date and time the test was created

This web application is developed from the perspective of the teacher who will be grading the students. The application is designed such that the teacher can create a new test and upload its answer key, upload the answer sheets or view student results.

1. *Creating a test* – Teacher has to enter the test name and its answer key. On submitting the form, the data is saved into the database with a unique id and the datetime created.
2. *Uploading the answer sheets* – The teacher can upload the answer sheet by browsing through his/her file explorer and can select all relevant images at once. The teacher also has to select the test, the answer key of which will be fed to the algorithm and the student will be graded accordingly.
3. *Viewing the result* – The teacher can select a test and fetch the data of all the students who attempted the test.

Fig 1. Homepage of the application. User can create new test, upload answer sheets or view results

Fig 2. Form to create a new test. Teacher has to enter the test name and its answer key.

Fig 3. Form to upload images. Teacher has to choose the test, the answer of key of which will be fed to the algorithm.

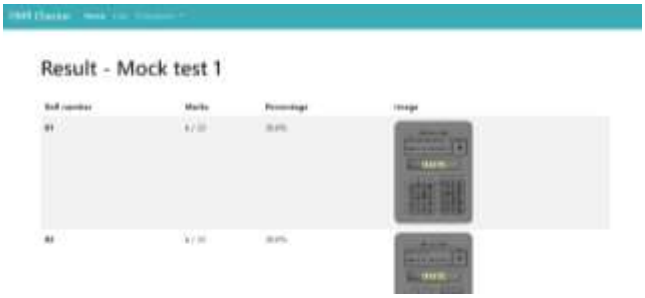


Fig 4. Results page after submitting all the images. The roll number is taken from the filename.

VI. Future Scope

1. Currently the algorithm presented in the paper does not take into account, if the user by chance does not mark an answer. A bubble is considered marked if its pixel value is above 4000. If none of the four bubbles of a question are above 4000, the question

can be considered unanswered. In such a case the algorithm can be modified to award 0 marks to that question.

2. At the front-end part the user, can be given options to configure the exam. For example, how many marks to award for each correct answer, option to set negative marking etc.
3. Each student is identified by his/her roll number while displaying the result. The roll number is taken from the filename. This can be improved to detect the roll number from the answer sheet itself using text detection.
4. Option can be given to the concerned teacher to edit the test answer key or test name.
5. Authentication and authorization can be implemented in the system.
6. The proposed system can also be deployed as an Android/iOS application to be used on mobile phones.

VII. Conclusion

This work presents a system for evaluating Optical Mark Recognition sheets. This project is expected to be used by small-scale institutions who can grade a large number of students efficiently and quickly. This system can also be used for other applications including questionnaires, surveys and forms. The only limitation of the proposed system is that the algorithm is highly dependent on the OMR sheet. Slight changes in the sheet may cause the algorithm to give false results. However, organizations using this system would be expected to use the template used in this paper. Furthermore, we may also have different templates for different number of questions. In that case, the algorithm would vary depending on the number of questions. The application can also be scaled up to be used by students to view their results and take the entire process online. Doing so, institutions will have access to a lot of data regarding students' performance. Hence, fast and reliable systems like this will enable teachers to save time, money and energy.

References

[1] Ismael Amara Belag, Yasemin Gultepe and Tark Melude Elmalti, "An Image Processing Based Optical Mark Recognition with the Help of Scanner", (IJIER), Volume 7, Issue 2, ISSN: 2277 – 5668.

[2] Rusul Hussein Hasan, Emad I Abdul Kareem, "An Image Processing Oriented Optical Mark Reader Based on Modify Multi-Connect Architecture MMCA", (IJMTER), Volume 02, Issue 07, [July– 2015] ISSN (Online):2349–9745 ; ISSN (Print):2393-8161.

[3] Zeki KÜÇÜKKARA, Abdullah Erdal TÜMER, "An Image Processing Oriented Optical Mark Recognition and Evaluation System", (IJAMEC), 2018, ISSN:2147-8228.

[4] Nirali V Patel, Ghanshyam I Prajapati, "Various Techniques for Assessment of OMR Sheets through Ordinary 2D Scanner: A Survey", (IJERT), Vol. 4 Issue 09, September-2015, ISSN: 2278-0181.

[5] Nalan Karunanayake, "OMR Sheet Evaluation by Web Camera Using Template Matching Approach", (IJREST), Vol. 2, Issue-8, Aug-2015.

[6] Nikita Kakade, Dr.R.C.Jaiswal, "OMR Sheet Evaluation using Image Processing", (JETIR), Vol.4, Issue-12, Dec-2017.