Automatic Answer Grader

INDIAN INSTITUTE OF TECHNOLOGY JODHPUR

SEMESTER IV AUGUST 2020

Guided By Dr. ANAND MISHRA

Submitted By :

Aditya Kumar	and	Kartik Vyas
(B18CSE002)		(B18CSE020)

Table of Contents

Goals	2
Tools and Techniques Learnt	3
Problem Statement	4
Development Plans and Execution	4
Morphology Operations	6
User Interface of the app	10
User Guidelines	12
Limitations and Future Plans	12
Result	13
Conclusion	14
Acknowledgments	14
References	14

Overview

- Since everyone cannot afford OMR grading machines to evaluate OMR sheets, we developed a solution for this.
- Mobile phones are very common nowadays, we have developed a mobile app to grade OMR sheets in very less time and with great accuracy.
- The idea was to build something that can grade the answer sheets in real time and is easy and convenient to use.
- With the help of this app and techniques used, instructors all over the world can grade students' answer sheets with a lot of ease.
- Everyone can easily use this app to hold exams, quizzes and competitions without the worry and hustle of concerns over evaluation.
- Also, the app is capable of grading irrespective of designs [only the questions and the responses need to be enclosed inside a rectangle]; this gives the instructors and academies the freedom to design their own answer sheets according to their convenience.

Goals

- To make evaluation easy for instructors and make error free system/platform for smooth conduction of exams everywhere and to remove the barriers of high costing investments in the OMR sheet grading machines and at the same time also enabling real time scanning and grading of the sheets.
- To design and implement algorithms aided by image processing libraries to develop the required functionalities and techniques to extract the responses of the student and validate them with the correct ones.
- To make an easy to use mobile app and which is supported by both, Android and iOS.
- The user can just click a photo in the app or can even upload photos, whichever option is more convenient.

- To test the app and platform enough to ensure accuracy and hustle free evaluation style.

Tools and Techniques Learnt

- **Image Processing Algorithms**
    - **Thresholding and Blurring**
    - **Erosion and Dilation**
    - **Sorting of Contours**
    - **Image Registration**
    - **Image Difference**
    - **Connected Components**
- **Mobile App Development**
    - **Tried to use Android Studio but then shifted to react native to build a cross platform application**
    - **Developed cross platform app in react-native**
    - **Used expo aided libraries and tools for several functionalities and testing**
- **Deployment**
    - **Developed API for python script using flask API**
    - **Hosted API on AWS EC2 instance**

Links

- **Codebase Link :**

  https://github.com/vyaskartik20/Automatic_Answer_Grader
- **Drive Link for App:**
  https://drive.google.com/file/d/1LKdwpzz9ns5oS7FubbkkjqCBhWywStpv/view?usp=sharing
- **Demo Video Link:**

  https://youtu.be/FJnn1yFqpJQ

Achievements

- Tested the app on MidSem 1 data of CS222 course OMR sheet responses of over 65 students. The results are quite accurate with an average marks difference of 0.22 marks.

❖ Implemented the algorithms in a fashion such that the user can design specialised OMRs and grade them as well with great accuracy.

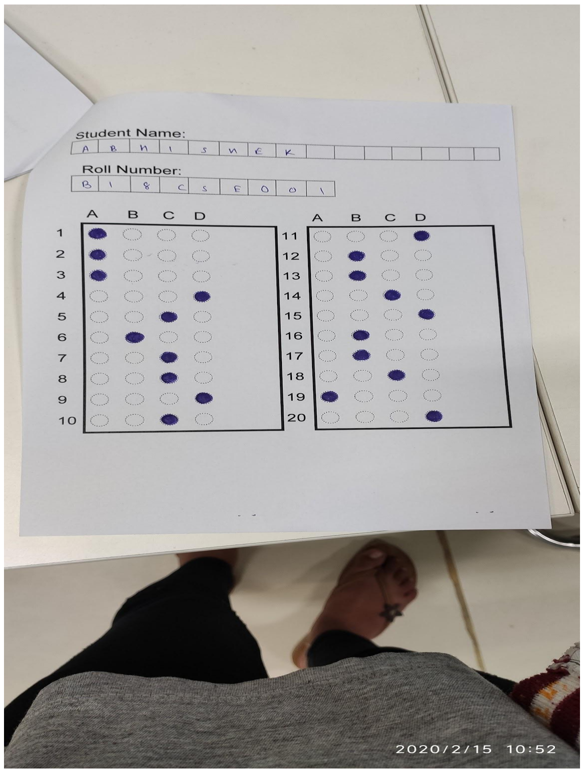
Problem Statement

Develop a solution for grading OMR Sheets using image processing techniques and build a mobile app that can grade OMR sheets by scanning/uploading photos in real time

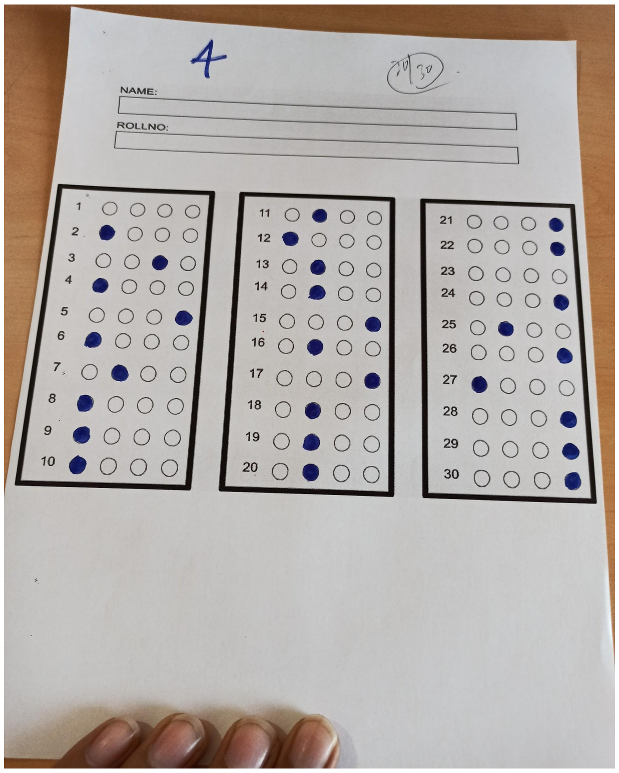
Development Plans and Execution

Data Collection

We were provided with data of 65 answer sheets by Dr. Anand Mishra to start our project in the right direction. We successfully tested our developed solution on these OMR sheets and observed great results. We also designed several OMRs and conducted tests to test the robustness of the design independence of the OMRs.



Mid Term Data



Self Collected Data

Techniques and Tools used

★ Image Processing

The Overall flow of our solution

- ❖ Cropping of rectangles [which contains the reponses] from the OMR sheets captured via the mobile app.
- ❖ Registering the student's response with respect to the ground truth and the blank OMR.
- ❖ Taking the difference of registered rectangles with rectangles of the ground truth and rectangles of the blank OMR.
- ❖ Calculating the number of connected components in both the differenced images and doing necessary maths for calculation of marks.

Step I : Cropping

I. Thresholding

Thresholding is a technique in using which we assign pixel values in relation to a particular value. In thresholding, each pixel value is compared with a threshold value. If the pixel value is less than the threshold, it is set to zero, otherwise, it is set to the maximum value. In Computer Vision, this technique of thresholding is done on grayscale images. So initially, the image has to be converted in grayscale color space.

II. Blurring

If a blurred image is observed carefully then a common thing to notice is that image is smooth meaning edges are not observed. A filter used for blurring is also called a low pass filter, because it allows low frequency to enter and stop high frequency. Here frequency means the change of pixel value.

Convolution **:** Image is denoted as a matrix inside a computer. An image contains a lot of features like edge, contrast etc. In image processing, features have to be extracted from the image for further study of image.

III. Hough Lines- Approach, Finding vertical lines and horizontal lines

Morphology Operations

Morphology is a set of image processing operations that process images based on predefined structuring elements known also as kernels.

Two of the most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of the object in an image, while erosion does exactly the opposite.

A) *Dilation*: The value of the output pixel is the maximum value of all the pixels that fall within the structuring element's size and shape.

B) *Erosion*: The vice versa applies for the erosion operation. The value of the output pixel is the minimum value of all the pixels that fall within the structuring element's size and shape.

IV. Filtering contours

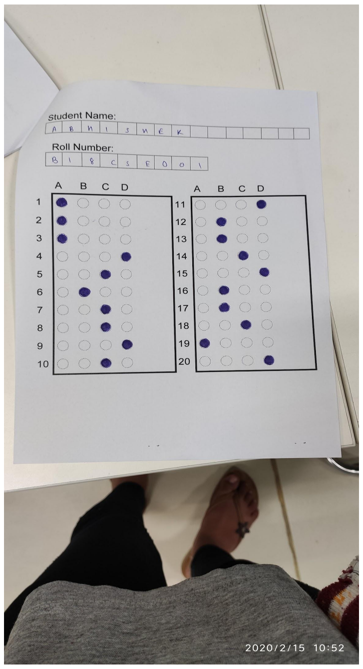
- ❖ Sort contours according to their size/area,
- ❖ Sort contoured regions from left-to-right and top-to-bottom.

V. Finding the Coordinates

We used Numpy library to convert all the coordinates of a contour into a linear array. This linear array would contain the x and y coordinates of each contour vertex. The key point here is that the first coordinate in the array would always be the coordinate of the topmost vertex and hence could help in detection of orientation of an image.

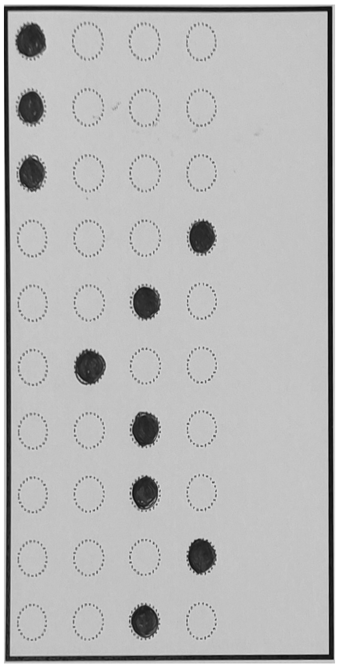
VI. Applying Perspective Transform and Extracting Rectangle

In Perspective Transformation, we can change the perspective of a given image for getting better insights about the required information. We need to provide the coordinates of the vertices of the contour for which the transformation is to be done

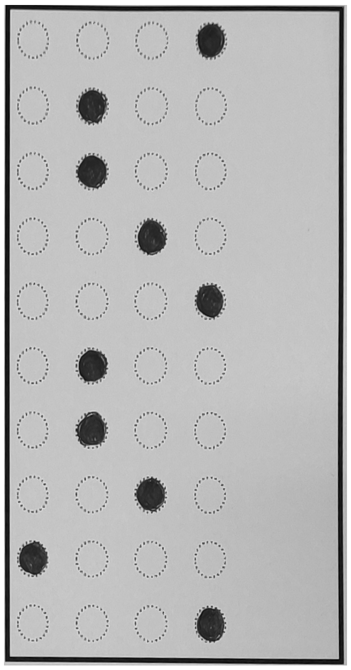


.

Image Captured by Phone



Rectangle 1



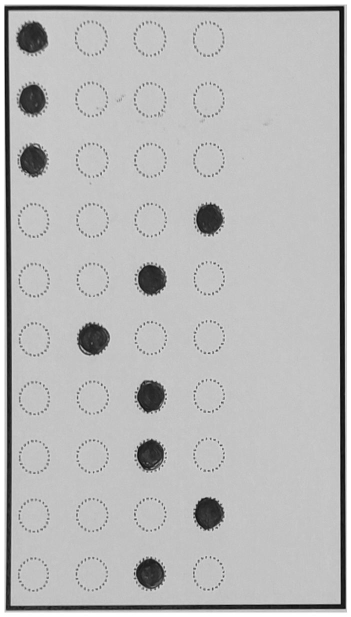
Rectangle 2

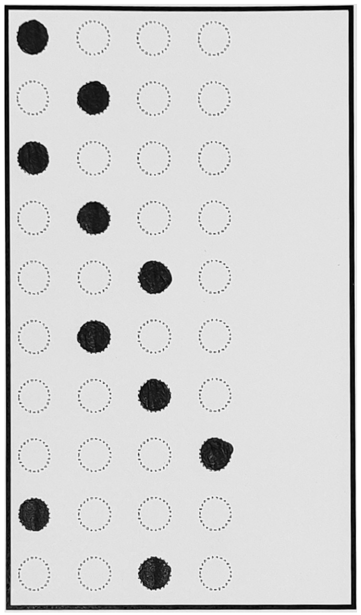
Step II : Registration

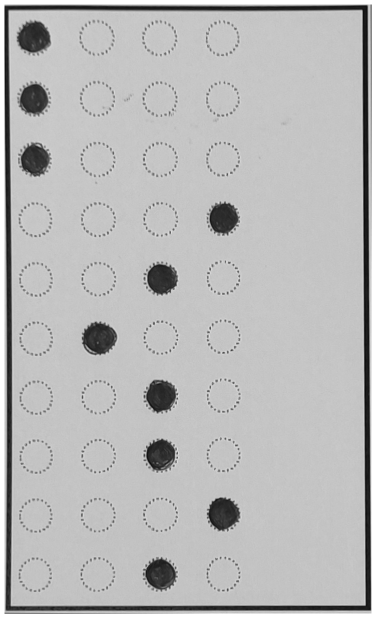
We first find the coordinates of the rectangles of the response sheet, answer key and the blank OMR. Image registration is an image processing technique which helps us align different images.

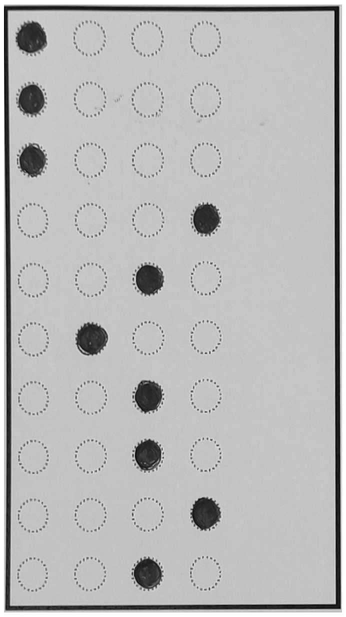
Alignment can be looked at as a simple coordinate transform. Our algorithm works as follows:

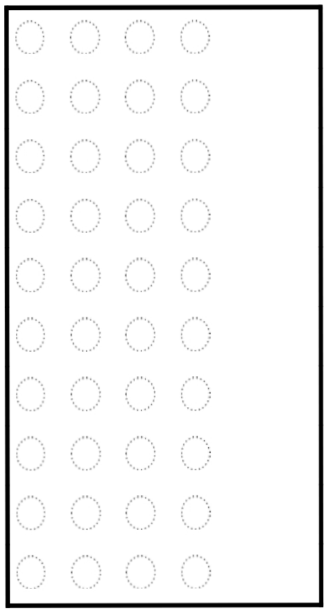
- Convert both images to grayscale.
- Find coordinates of vertices of extracted rectangles
- We are using these coordinates of rectangles as key points for matching between both images
- Using these key points of the two images we find homography transform
- Apply this transform to the original unaligned image to get the output image.

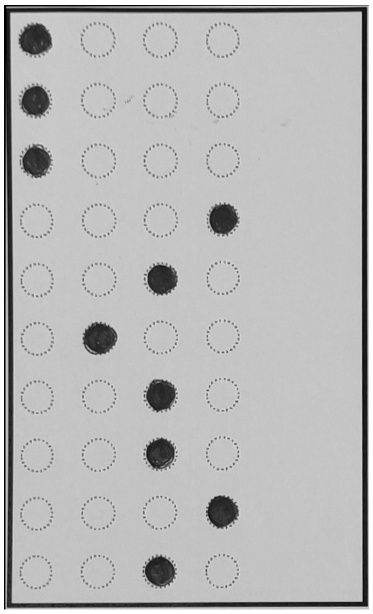

Cropped Image


Corresponding image from ground truth for Registration


Registered Image

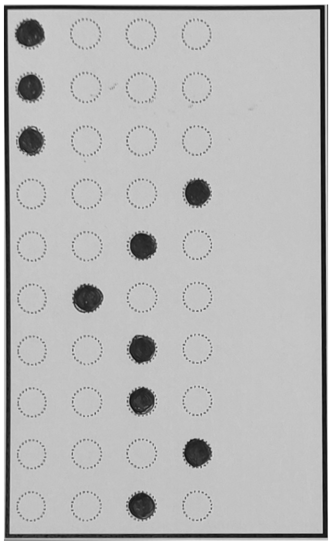

Cropped Image

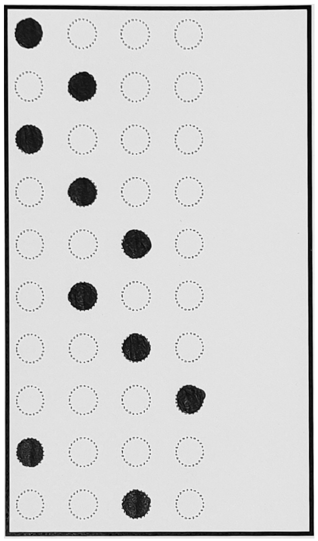

Corresponding Image from empty OMR for registration

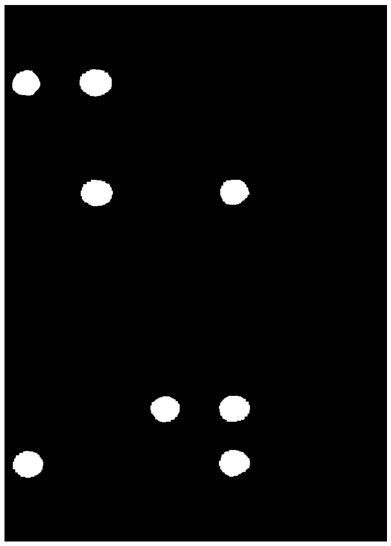

Registered Image

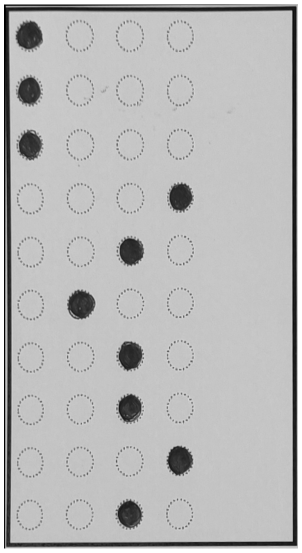
Step III : Difference

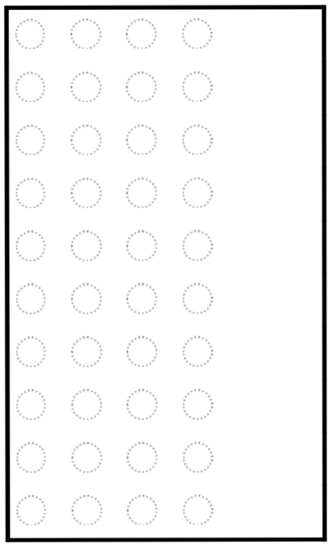
- Taking the difference of the response sheet with respect to answer key and black OMR sheets.

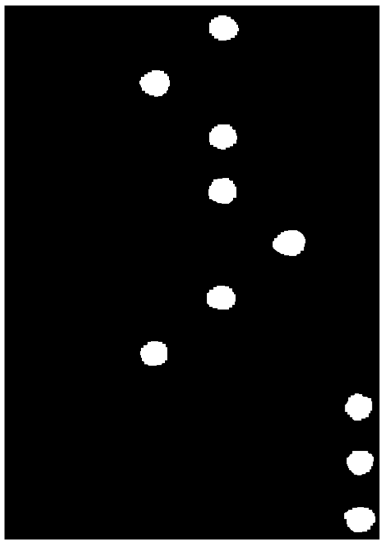

Registered Image


Rectangle of Ground Truth


Difference of both Images

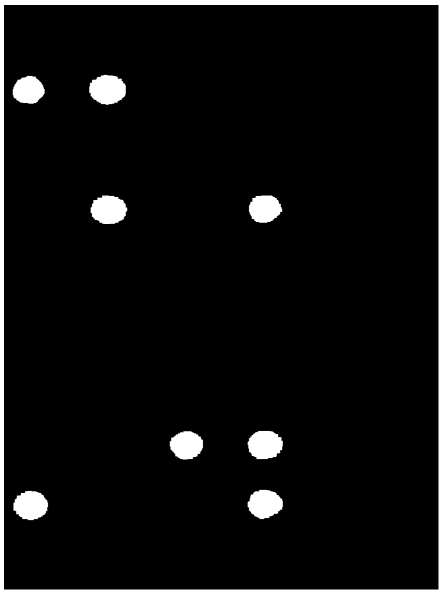

Registered Image

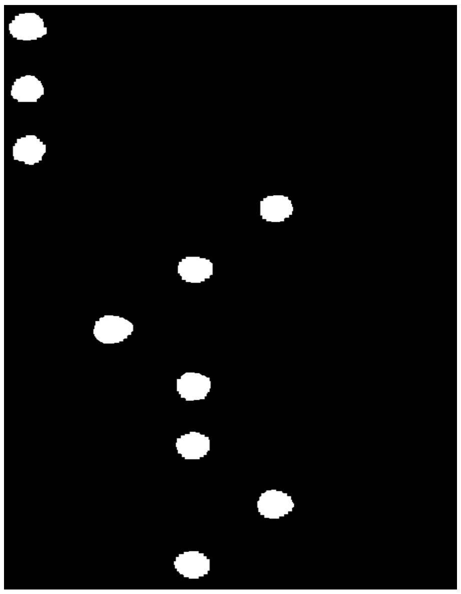

Empty Error


Difference of both Images

Step IV : Connected Components

- ❖ Using connected components, finding the number of responses in both the cases.
- ❖ Doing necessary math, to compute the final marks


Differenced Image with ground truth

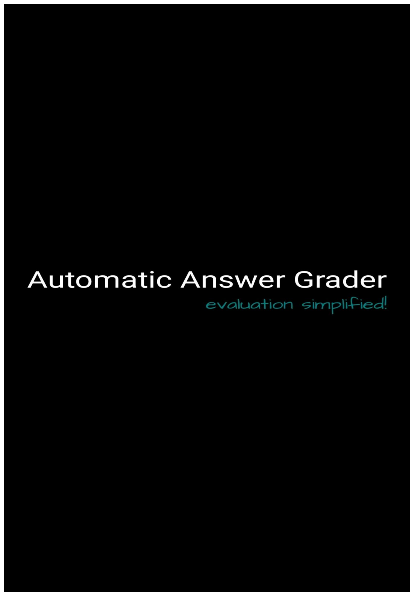

Differenced Image with empty OMR

★ Mobile App development

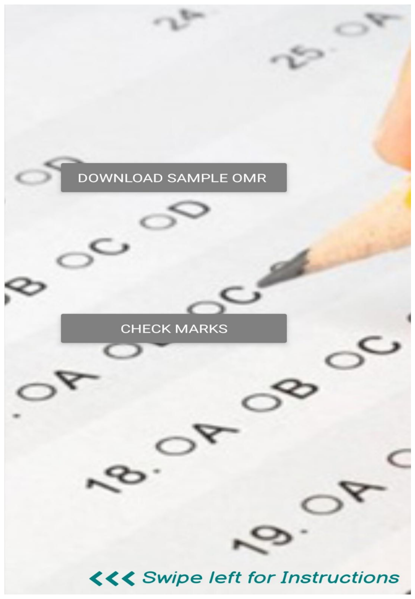
Firstly, we tried to build the app using Android Studio.

Then to increase the dynamiticity and to add more value to the user experience, we shifted to React Native. This helped us in building a cross platform app i.e it can be used by both Android and ioS users. We also used expo tools and libraries to add several functionalities and for testing and debugging of the app.

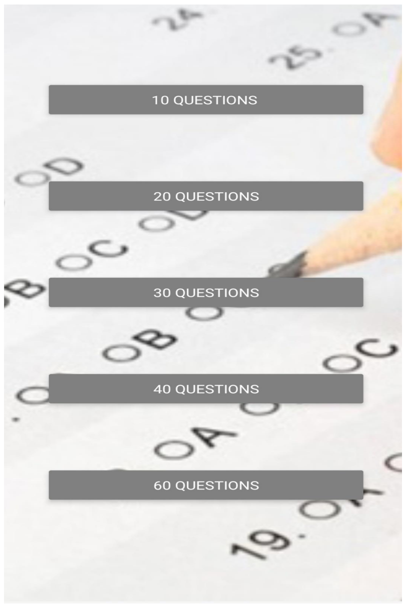
User Interface of the app



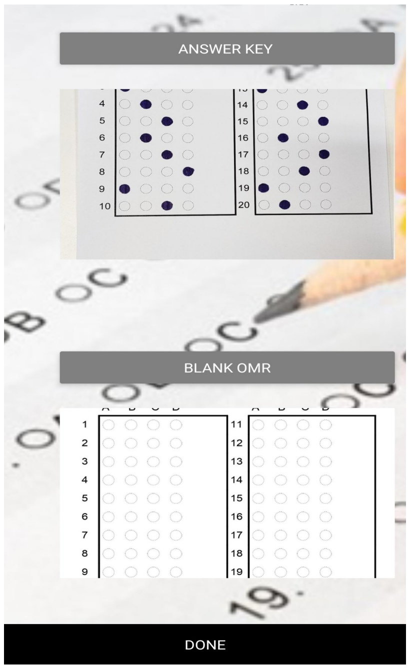
Home Screen



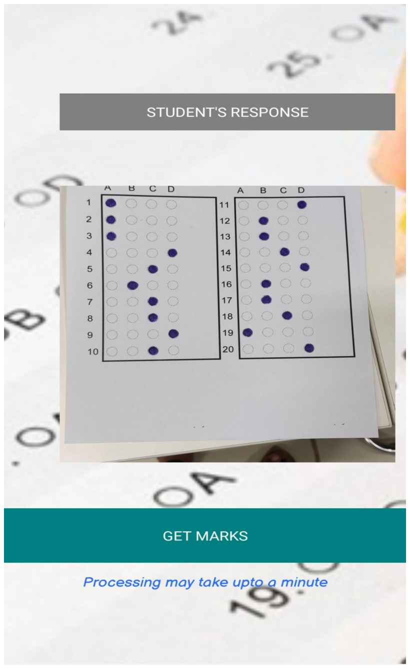
Option to download OMR and check marks



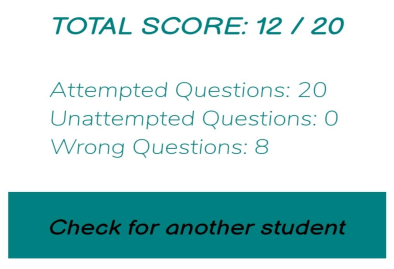
Option to download different types of OMR



Capturing ground truth and blank OMR



Capturing student's response


TOTAL SCORE: 12 / 20

Attempted Questions: 20
Unattempted Questions: 0
Wrong Questions: 8

Final Marks Displayed

★ Deployment

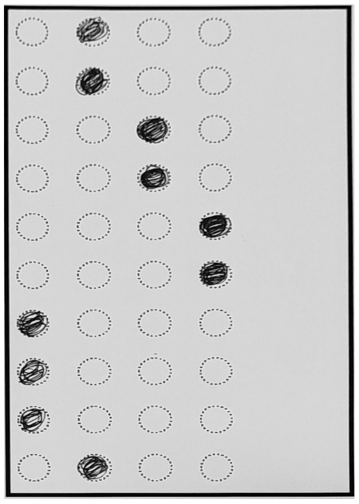
- ❖ Developed API for python script using flask API
- ❖ Hosted API on AWS EC2 instance

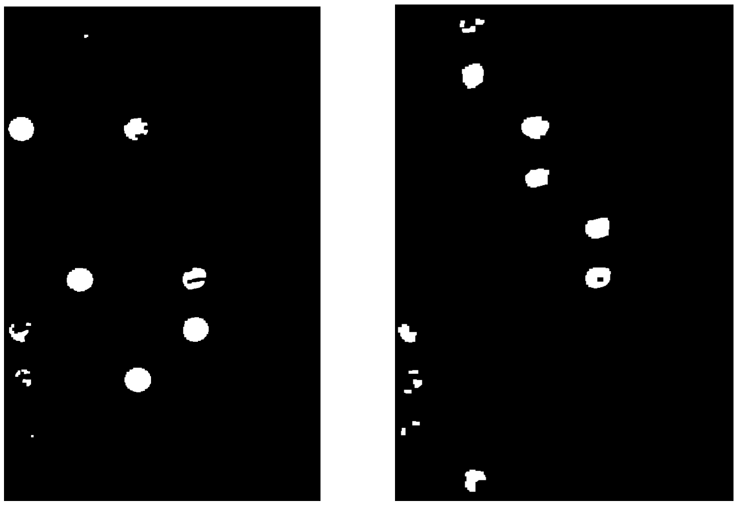
User Guidelines

- Following are the guidelines that the user must adhere to get the most accurate results:
- Take the photo in considerable amount of light
- The photo should not have any shadow, position your hands accordingly while taking the photo
- The photo of the OMR should be vertically upside down; however, tilting or photos at an angle would not be a problem. Photos in horizontal orientation of the OMR should not be used.
- After sending a request of getting marks, a data transfer of roughly 10 to 15 MBs will take place. So be patient according to your network connection. A good network connection is to be preferred.
- Three images are needed to be uploaded.
    - The answer key
    - The blank OMR sheet that is used for examination
    - The student's response that is to be evaluated
- It is recommended to use the OMRs provided in the app. Though any OMR sheet which contains the questions inside rectangular boxes (like the provided OMRs) can be graded accuarately.

Limitations and Future Plans

In case the student fills the OMR response sheet in such a loose fashion as shown below, the difference with respect to answer key and blank OMR would not be optimal and there would be error in counting of response during the connected components technique.





Improper marking of OMR

Improper differenced images

In the future, multiple correct answer questions can be graded and detecting which questions were attempted incorrectly and the unattempted questions can also be added.

Result

We tested the app on the Mid Sem I data of CS222 course [the exam was taken on OMR sheets]. There are 65 answered OMRs. Along with this, we verified results on several designed OMRs to test the robustness of the app. The following is the result observed on the mid sem data

- 0.22 marks difference on average.
- 53 results were perfect
- 5 results had difference of .5 marks
- 1 results had difference of 1 marks
- 2 results had difference of 1.5 marks
- 4 results had difference of 2 marks

Thus the solution is working as it was expected to be.

Conclusion

We successfully designed and implemented algorithms aided by image processing libraries and built the mobile app in React Native and deployed it on AWS EC2 instance.

The app can be used to grade the answer sheets of a large scale as well as small scale examination. The app gives fairly accurate results in a reasonably less amount of time.

Acknowledgments

We are thankful to our instructor, Dr. Anand Mishra, for providing us the necessary guidance, without which this app could not have been completed.

Also, we would like to thank Abhirama Subramanyam Penamakuri for his much needed support and helping us throughout the course of the project.

References

- https://www.pyimagesearch.com/2014/08/25/4-point-opencv-getperspective-transform-example/
- https://www.geeksforgeeks.org/find-co-ordinates-of-contours-using-opencv-python/
- https://www.pyimagesearch.com/2015/04/20/sorting-contours-using-python-and-opencv/
- https://reactnative.dev/docs/0.5/images
- https://www.geeksforgeeks.org/image-registration-using-opencv-python/
- https://www.pyimagesearch.com/2017/06/19/image-difference-with-opencv-and-python/

- https://www.geeksforgeeks.org/perspective-transformation-python-opencv/#:~:text=In%20Perspective%20Transformation%2C%20%2C%20we%20can,insights%20about%20the%20required%20information.
- https://docs.opencv.org/3.4/d7/d4d/tutorial_py_thresholding.html
- https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html
- https://docs.opencv.org/3.4/d3/dc0/group__imgproc__shape.html
- https://reactnative.dev/docs/0.5/components-and-apis
- https://www.geeksforgeeks.org/python-build-a-rest-api-using-flask/
- https://docs.aws.amazon.com/ec2/?id=docs_gateway