Grade Auto Filler

Team Members

Name	Sec	BN
Ahmed Alaa	1	7
Beshoy Morad	1	20
Zeyad Tarek	1	28
Waleed Hesham	2	36

Project Needs

1- Python 3.10
2- OpenCV
3- NumPy
4- Skimage
5- Imutils
6- PyTesseract
7- Xlsxwriter

Project Description

The main idea of (Grade Auto Filler) is to give an image of a table with some data to the program and get an output of excel sheet containing the data that was in that image after mapping the symbols to the wanted grades.

The main idea of (Bubble sheet correction) is localize the filled circles referenced to the number of columns and number of rows and compare them to a given model answer and provide sheet of students' grades.

Constraints

1- The image should be clear, and the light distribution is appropriate and not too bad.
2- Handwritten numbers and symbols should be clear.
3- Bubble sheet must consist of 2 columns or 3 columns only.
4- The paper should be fully visible.
5- The background of the paper should be dark.
6- The paper should be flat (not bent).
7- Use the KNN to get the ids in case of high-resolution images only, otherwise use the Hybrid method or the OCR. (Hybrid is getting the best results in all cases)
8- ID is detected through the specific bubbles, so if the student mark wrong bubbles I will not correct them for him.
9- You must provide the file of IDs of all student papers if not exist in the paper itself and rename the images by the students' IDs.

Used algorithms

[Module 1]: Grade Auto Filler

To detect cells with normal image processing:

1. Enhance the image of the cell to remove the noises
2. Get the angle of lines in that cell and if it was in a certain range then this cell is **correct mark**
3. Perform opening operation on the cell with vertical and horizontal kernels to get the number of vertical and horizontal lines in that cell.
4. Get the contours and calculate the area then check if it was higher that a certain value then this cell is **Box**
5. Perform Hough Circles with a certain value for min and max radius to check if that cell is **Question mark**
6. If it was nothing from the above, then check if it was **horizontal lines or vertical lines** cell
7. If it was not horizontal or vertical, then it is an empty cell

To detect cells using HOG and KNN:

1- We have to train the HOG, KNN models using a set of images with different scales and different shapes.
2- We can use that model to predict the upcoming cells.

To extract cells:

1. Apply gaussian blur + canny edge detection
2. Get contours of the edged image and sort them by area and select largest one (paper)
3. Apply four-point transform to this contour to correct skewness
4. Apply morphological opening using row structure element to remove anything except horizontal lines
5. Apply Hough Lines transform to detect horizontal lines and remove unwanted lines (not perfectly horizontal)
6. Apply step 4, 5 again for vertical lines
7. Draw the detected lines in a fully white image with same size of original one
8. Get contours of this image and sort them from top to bottom
9. Filter contours based on its size to eliminate false cells
10. Extract cells from original image from these contours' coordinates

[Module 2]: Bubble Sheet Auto Corrector

1. Apply skew correction which is implemented in the previous module (Module 1).
2. Crop the student's information ticket.
3. Convert the image into grayscale.
4. Apply Canny Edge Detector.
5. Detect all the circles (whatever filled or not) using Hough Circle, which is represented in center (a, b) and radius (r).
6. Sort the circles according to Y values.
7. Sort each row's circles according to X values.
8. Resolve each question's circles and take rectangle over each circle and binarize it.
9. If the number of white pixels (value = 255) is less than specified threshold value, then this circle is classified as filled pixel.
10. If the student's ID is included, then separate the 1st 10 elements and extract the ID from it by comparing it by the choice's orders.
11. Compare the detected answers with the model answer, then create a list of True/False.
12. Send a JSON object containing "id", and "answers"
13. Use the JSON object to generate the excel sheet to summarize the results of the student.

[BONUS] Extra Feature:

1. Create [configBubble.conf] to customize your own constraints on the model like (if there is ID or not) or (number of questions, number of choices, etc.)
2. Load students' IDs from file.
3. Export the autocorrected answer paper on the hard disk.
4. Can handle a lot of student's papers with same type of samples and one model answer using one click.

Experiment results

We used 15 samples with different angles of capturing (Skewing, orientation, scale) and with different hand-writing fillings.

The results:

Sample Number	Number of wrong detected symbols with normal image processing	Number of wrong detected codes with OCR	Number of wrong detected numeric values with OCR	Number of wrong detected numeric values with features + classifier
1	1	0	2	2
2	0	0	4	3
3	2	1	4	2
4	1	0	5	1
5	1	0	1	0
6	1	3	1	1
7	0	0	1	2
8	0	1	1	0
9	1	1	6	0
10	0	0	4	1
11	0	0	7	0
12	0	0	2	1
13	0	0	4	0
14	0	0	2	0
15	0	0	2	0

Accuracy of Symbols

Number of test cases = 17 cell * 2 columns * 15 samples = 510

Number of wrong detected symbols = 7

Accuracy of symbols detection **with normal image processing techniques**

= (510 - 7) / 510 = **98.6%**

Accuracy of symbols detection with HOG model = **100%**

Accuracy of code and numeric values detection using OCR

Number of test cases = 17 cell * 15 sample = 255

Number of wrong detected codes = 6

Number of wrong detected numeric values = 46

Accuracy of code detection = (255 - 6) / 255 = **97.6%**

Accuracy of numeric values detection = (255 - 46) / 255 = **82%**

Accuracy of code and numeric values detection using features and classifier

Number of test cases = 17 cell * 15 sample = 255

Number of wrong detected numeric values = 13

Accuracy of numeric values detection = (255 - 13) / 255 = **94.9%**

Accuracy of Bubble sheet Correction

Number of samples = 2

Number of questions = 70

Number of wrong detected questions = 0

Accuracy of numeric values detection = (70 - 0) / 70 = **100%**

Analysis

After testing the program with a lot of samples we noticed that OCR is good with codes but not very efficient with numeric values, also most of wrong numeric values detected was (**digit 1**) and sometimes (**digit 7**), but the KNN model is very good with the handwritten numeric values and not very efficient with the codes, so we decided to make a hybrid model to use the OCR to get the codes and the KNN to get the handwritten numeric values.

For symbols, most of wrong detected cells were right mark and question mark because the angle of lines of right marks sometimes gets out of the specified range, also question marks sometimes the radius of the circle gets out of the specified range.

Workload

Name	Workload
Ahmed Alaa	Bubble sheet module and make the config file.
Beshoy Morad	Detect the symbols using pure image processing and using features and classifier, detection of numeric values and codes with OCR, and design the config file.
Zeyad Tarek	Detection of numeric values and codes with features and classifiers (KNN), collect datasets, and train the model
Waleed Hesham	Image enhancements, cell extractions, and helped with bubble sheet module.