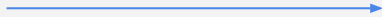
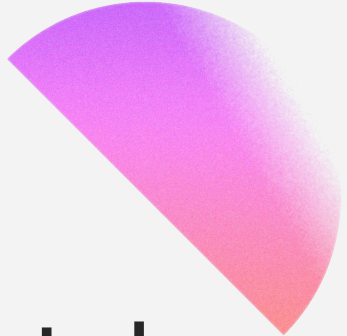


MCQ Answer Script Marking (OCR)





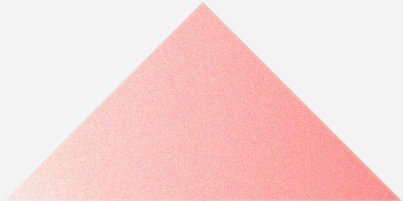
What will be presented

- Project brief and objectives
 - Methodology
 - Results
 - Conclusion
- 



The project and its objectives

Project brief

- Python and OpenCV based project.
 - Program accepts a picture of an answer script.
 - Detect rectangles and then marks on the selected areas.
 - Gives the marks out of 100%.
 - Marks the correct and wrong answers on the image.
- 

Objectives

- Try to implement a OpenCV based real-time MCQ marking system design
- Reduce the time takes to markup the answer scripts
- Reduce the mistakes that can happen when marking manually
- Can use in any laptop / device that can run python

METHODOLOGY

Common operators

- Grayscale conversion, resizing the image, and imshow (show output image)

Canny operator

- `imgCanny = cv2.Canny(imgBlur, 10, 70)`
- Canny operator used to identify the edges main parts of the image.
- Because Canny uses hysteresis thresholding and non-maximum suppression, the resulting edges are usually thinner and more linked.

METHODOLOGY

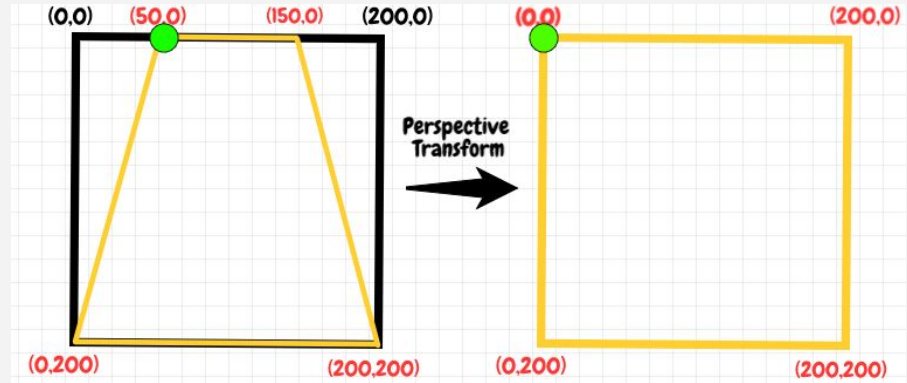
Find contours

- `contours, hierarchy = cv2.findContours()`
- Find all the available contours
- `rectCon = tools.rectContour(contours)`
- Find and get array of all the rectangular contours

METHODOLOGY

Warp transformation

- `cv2.getPerspectiveTransform()`
- `cv2.warpPerspective()`
- `cv2.getPerspectiveTransform()` -> use to get transformation matrix
- `cv2.warpPerspective()` -> apply warp transformation
- Warp transformation use to change the image shape



METHODOLOGY

Threshold

- `cv2.threshold()`
- Apply threshold on selected area of the images.
- Make it easy to identify the OCR marks on the image

Add weighted

- `cv2.addWeighted()`
- Use to multiply image with original image

Results

Canny Operator

IMAGE PROCESSING
OCR TEST

1.	0	0	0	0	0	11.	0	0	0	0	0
2.	0	0	0	0	0	12.	0	0	0	0	0
3.	0	0	0	0	0	13.	0	0	0	0	0
4.	0	0	0	0	0	14.	0	0	0	0	0
5.	0	0	0	0	0	15.	0	0	0	0	0
6.	0	0	0	0	0	16.	0	0	0	0	0
7.	0	0	0	0	0	17.	0	0	0	0	0
8.	0	0	0	0	0	18.	0	0	0	0	0
9.	0	0	0	0	0	19.	0	0	0	0	0
10.	0	0	0	0	0	20.	0	0	0	0	0

Total Marks:

Results

Contour Detection

IMAGE PROCESSING
OCR TEST

1.	●	○	○	○	○	11.	●	○	○	○	○
2.	○	●	○	○	○	12.	○	○	●	○	○
3.	○	○	○	●	○	13.	○	○	○	○	●
4.	●	○	○	○	○	14.	○	●	○	○	○
5.	○	○	○	○	●	15.	○	○	●	○	○
6.	○	○	●	○	○	16.	○	○	○	○	●
7.	○	●	○	○	○	17.	●	○	○	○	○
8.	○	○	○	●	○	18.	○	●	○	○	○
9.	●	○	○	○	○	19.	○	○	○	●	○
10.	○	○	○	○	●	20.	○	○	○	●	○

Total Marks: XXXXXXXXXX

Results

Selecting the main rectangles

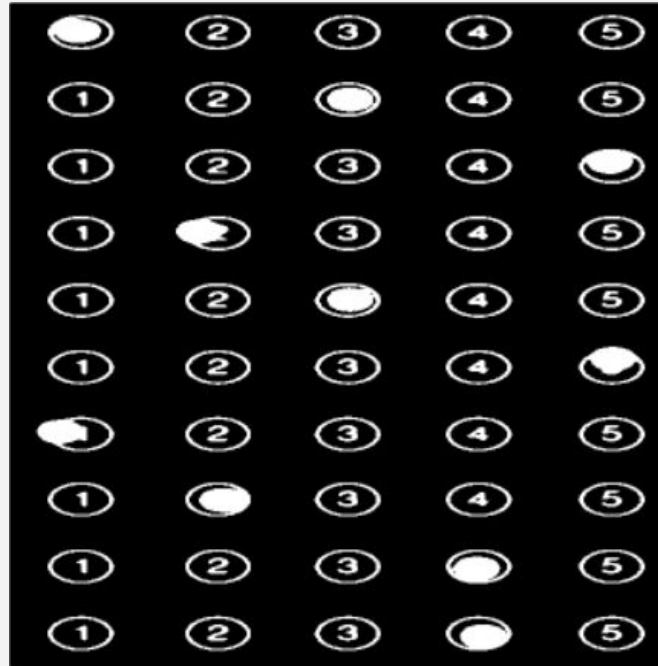
IMAGE PROCESSING
OCR TEST

1.	●	2	3	4	5	11.	●	2	3	4	5
2.	1	●	3	4	5	12.	1	2	●	4	5
3.	1	2	3	●	5	13.	1	2	3	4	●
4.	●	2	3	4	5	14.	1	●	3	4	5
5.	1	2	3	4	●	15.	1	2	●	4	5
6.	1	2	●	4	5	16.	1	2	3	4	●
7.	1	●	3	4	5	17.	●	2	3	4	5
8.	1	2	3	●	5	18.	1	●	3	4	5
9.	●	2	3	4	5	19.	1	2	3	●	5
10.	1	2	3	4	●	20.	1	2	3	●	5

Total Marks: XXXXXXXXXX


Results

Warp and threshold



Results

Grid and mark answers

Results

Final result

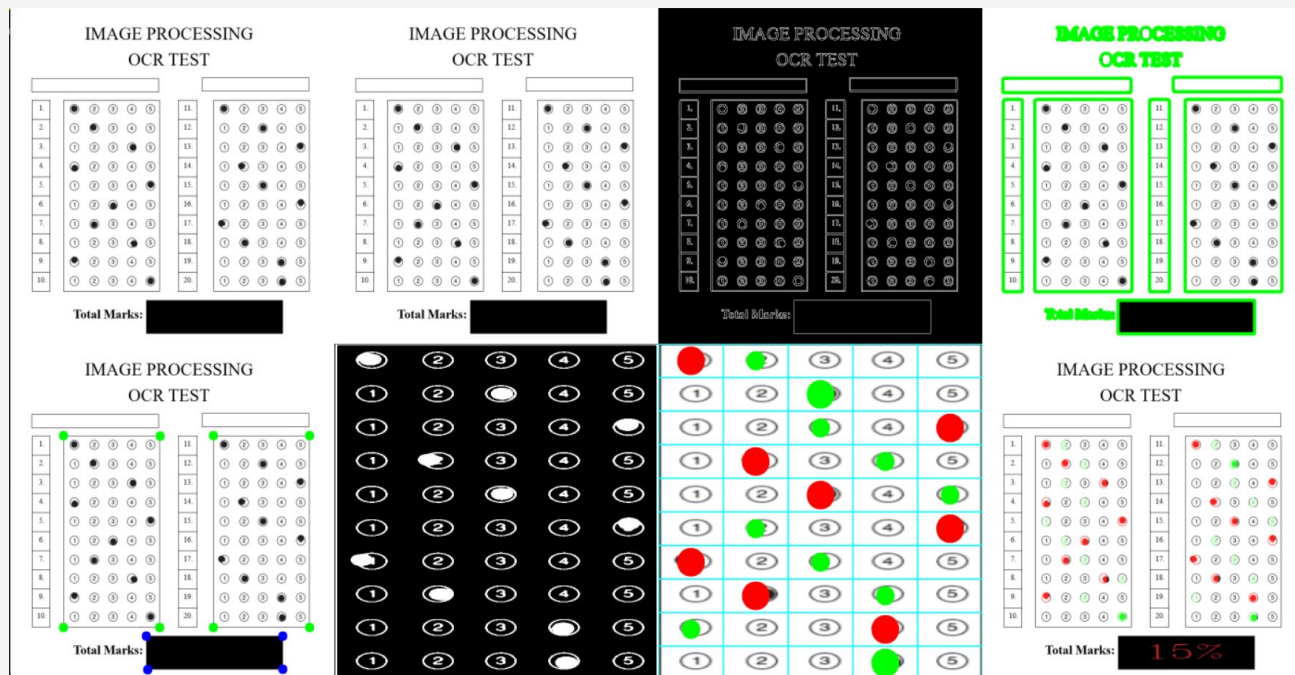
IMAGE PROCESSING
OCR TEST

1.	1	2	3	4	5	11.	1	2	3	4	5
2.	1	2	3	4	5	12.	1	2	3	4	5
3.	1	2	3	4	5	13.	1	2	3	4	5
4.	1	2	3	4	5	14.	1	2	3	4	5
5.	1	2	3	4	5	15.	1	2	3	4	5
6.	1	2	3	4	5	16.	1	2	3	4	5
7.	1	2	3	4	5	17.	1	2	3	4	5
8.	1	2	3	4	5	18.	1	2	3	4	5
9.	1	2	3	4	5	19.	1	2	3	4	5
10.	1	2	3	4	5	20.	1	2	3	4	5

Total Marks: **15%**

Conclusion

Final result



Conclusion

Challenges

- When the image not properly scanned, it's difficult to identify contours
- Moving from MATLAB to python with openCV.



Thanks!

Any questions?