Report

Description:

Program for grading multiple choice questions answer sheets. The program takes three input arguments, the answer sheet to be read, the output image name to indicate the marked answers by highlighting them and a text file name to print the marked answers.

Run-time: 4-5 mins on burrow server.

Algorithms used:

1. Canny Edge detection.
    - Sobel operator (gradient calculation)
    - Non-maximum suppression
2. Hough Transform

Implementation:

1. Our code runs on greyscale images, so we are checking if the input image is in greyscale, if not converting it to greyscale.
2. **Canny Edge detection:**
    - We are using the standard sobel operator to calculate the gradient magnitudes and directions.
    - Before performing non-maximum suppression we are using a sharpening filter to sharpen the image as we observed the edges were more prominent after sharpening.
    - We are then performing non-maximum suppression to calculate the actual edge pixels, by checking the gradient angles and the magnitude of nearby pixels
3. **Hough Transform:**
    - Our goal is build a grid around on the answer sheet so as to detect the location of the answer choices in the image.
    - In-order to do that we are generating lines along borders of the boxes using hough transform.
    - To perform this we are using the polar coordinate system, with rho and theta values for various lines.
    - Horizontal and vertical lines will have theta values 90 and 0 degrees respectively and rho values ranging along the width and height.
    - We made each pixels vote for various rho values which ultimately transform to a line in Cartesian coordinate system.
    - The rho values which were above a given threshold are selected to form lines of the grid.
4. Post this we have written logic based on some intuition from the given image to locate the boxes on the grid.

5. We are checking the ratio of total number of marked pixels in a box to the total number of pixels to decide if a student had marked that answer or not.

Challenges and Design decisions:

1. After using sobel operator and performing non-maximum suppression, the edges were not being detected properly. We tried a few Gaussian filters with different sizes and sigma values however no major improvement was visible. Finally when a sharpening filter was applied the edges became prominent.
2. During canny edge detection we are only checking edges which indicate a gradient change along 0 degrees or 90 degrees, as we are concerned with horizontal and vertical edges.
3. Also during hough transform we are performing voting only for lines with theta values 0 and 90 degrees for the same reason.
4. Each image required a different threshold for detecting all the lines of the grid. Hence we wrote logic to vary the threshold from image to image automatically depending on the number of lines detected with the initial threshold and keep changing the threshold until required number of lines are detected.
5. Canny edge detection and hough transform lead to two lines for each edge, which had to be eliminated.
6. Sometimes the grid formed is not perfect (like a few missing lines or wrong line) especially towards to lower half of the image, if the image has lot of noise. In that case we have written a crude logic to rectify the errors by approximating the distances between the boxes and their sizes. This can sometimes cause the program to crash.

Accuracy:

1. The current accuracy of the program is 90% – 99 % on the given images.

Limitations:

1. The program might or might not fail on new images because of the crude logic to rectify the errors in the grid.
2. However, the accuracy should remain high in case if does not crash.
3. We can ignore the crude logic and let the program run without errors, but the accuracy might drop to 50%-60%.
4. Again for the same reason, if the images vary a lot in terms of orientation, the program might not work

References:-

1. Course Materials.
2. Canny Edge detection Logic : http://www.aishack.in/tutorials/canny-edge-detector/
3. Hough Transform : https://alyssaq.github.io/2014/understanding-hough-transform/