Digital Image Processing

Project Report:

Optical Mark Recognition

Group Members:

Umer Mukhtar – i20-0696

Saad Farooq – i20-0555

Section: A

Contents

Project Description

Optical Mark Recognition – OMR, is an electronic method of gathering human-handled data by identifying certain markings on a document. The data can be surveys, MCQs, attendance etc. This project is only limited to MCQs but can be extended for other purposes with relevant modifications.

OMR, in this project, finds markings of the filled MCQs. It then identifies the corresponding option that was checked e.g. A, B, C, D … It then evaluates it according to the solution it is fed initially and calculates the score.

Project Working

From the GUI, the user initially selects the option for submitting the solution of the MCQ sheet. The user scans the solution sheet using a camera. From that sheet, OMR generates the answers in the form of a list.

Again, from the GUI, the user may choose the option to evaluate an answer sheet. The user scans the answer sheet using a camera and the OMR calculates the sheet's score and displays it to the user.

Libraries

Opencv

It is used for many functions of opencv mentioned later.

Numpy

It is used to process the images using multidimensional arrays.

Time

Sys

Functions

arrangePoints()

opencv: inRange()

It is used to calculate output image with respect to the hue, value, saturation values given to it. In this case, it is used to create a boundary around the 'page-region' in the scanned picture.

opencv: Canny()

It is used to find the edges of the paper using Canny Edge Detection.

opencv: findContours()

This function finds contours of the image i.e. any boxes, bounded-shapes etc. present in the image. Initially it is used to get boundaries of the image. Later, again used to get the boundaries of the boxes containing the MCQs.

get_top_down()

This function is used to warp the image correctly to fit the screen keeping the corners in correct positions for warping. It uses getPerspectiveTransform() and warpPerspective() functions of the library opencv.

opencv: getPerspectiveTransform()

This function transforms the image such that it's co-ordinates are mapped into new co-ordinates that are provided as arguments.

opencv: warpPerspective()

After using getPerspectiveTransform(), warpPerspective() warps the image such that it is stretched across multidimensional NumPy array.

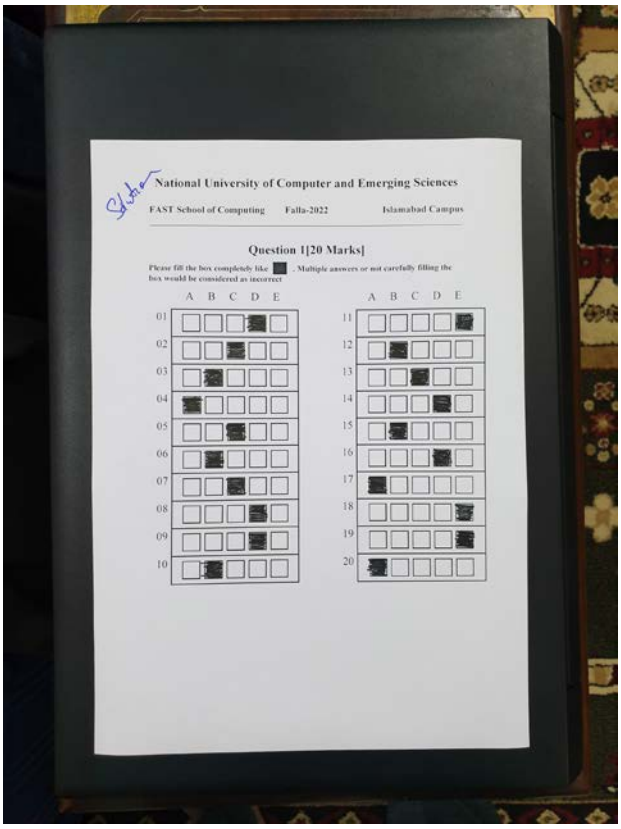
largestCC()

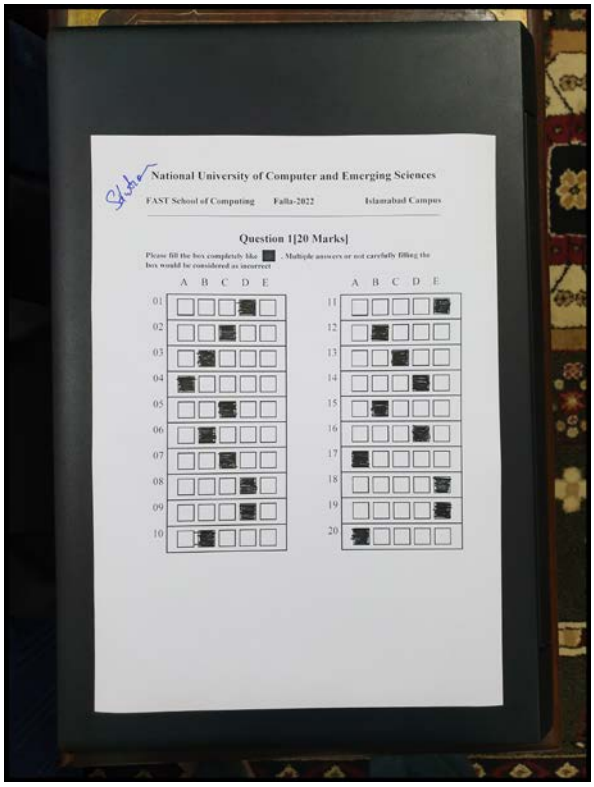
This function is used to extract the largest connected component present. It is used in this project to detect filled boxes of MCQs.

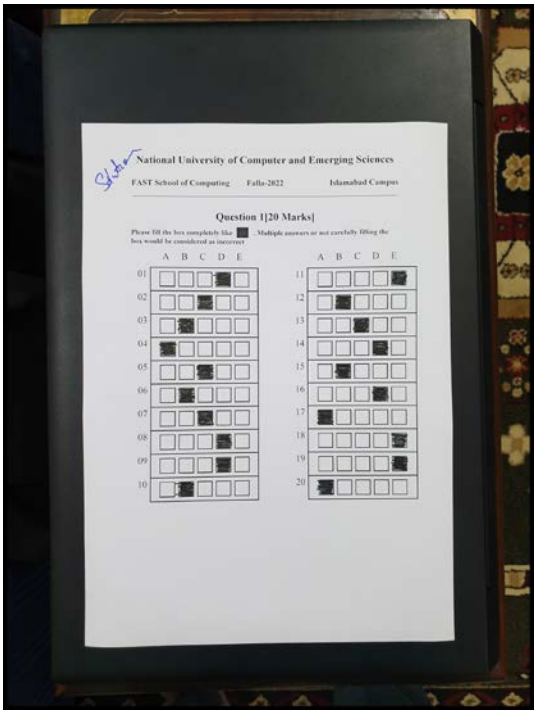
opencv: videoCapture()

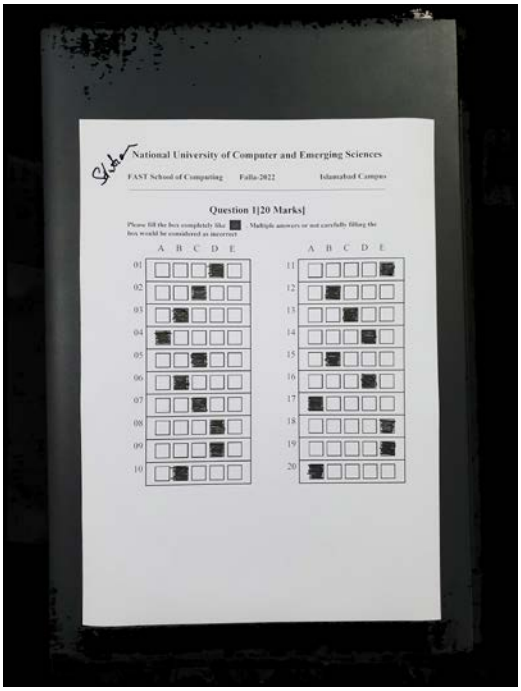
This function is used to capture the frames that the camera captures using video.

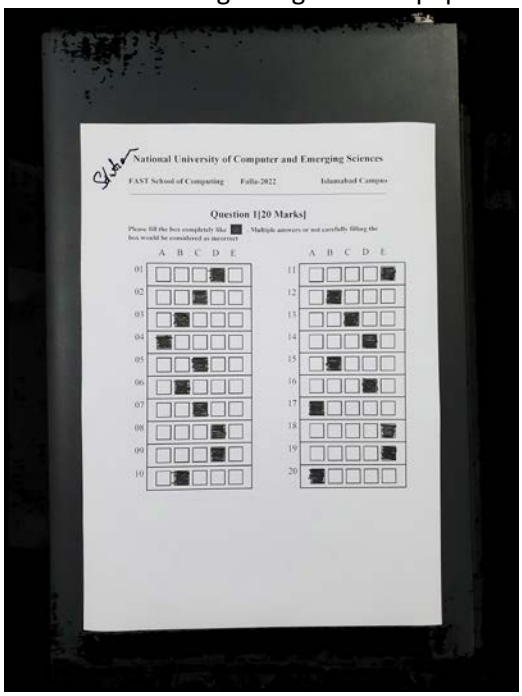
Working

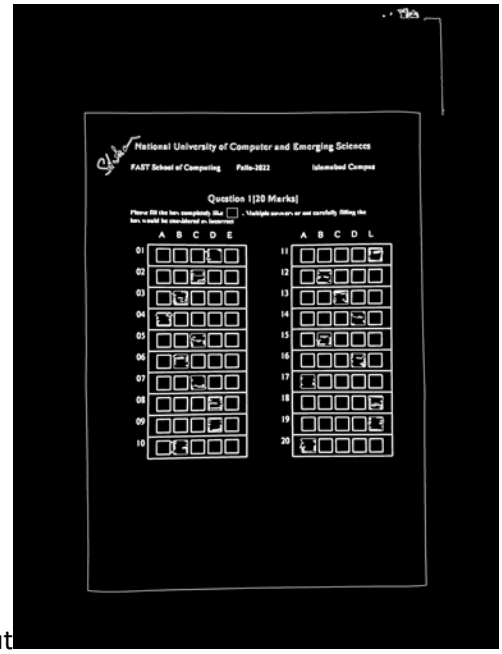
- Initially, the image is padded with black pixels, in-case a (non-MCQ) part of the paper is missing from the image.
  - Input: 

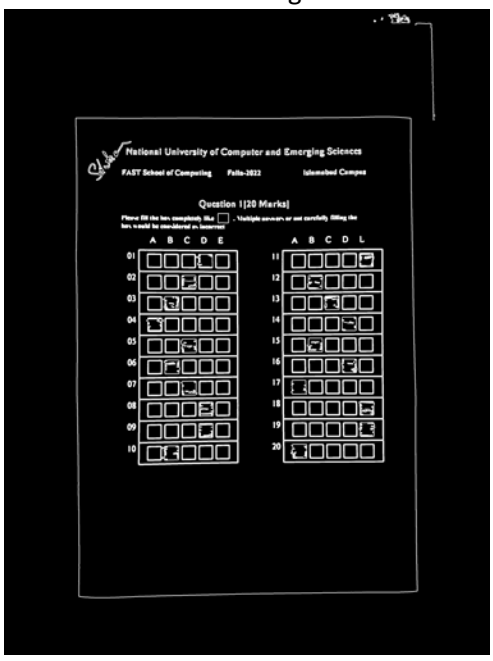
- Output: 

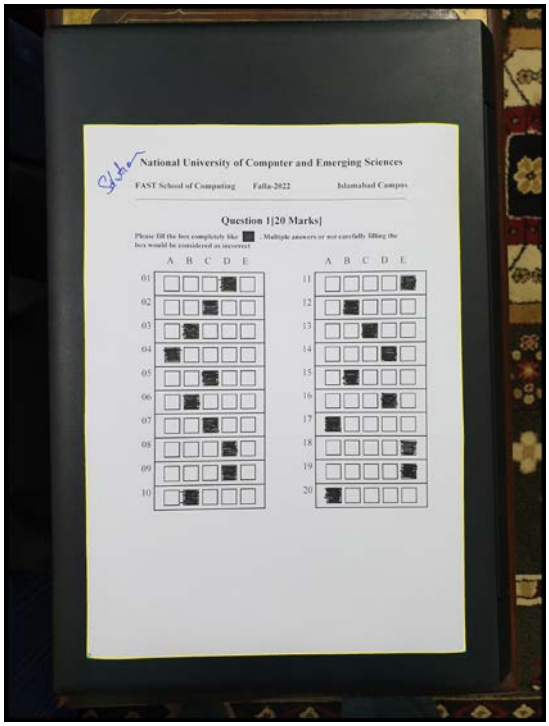
- Initially, the image may contain objects other than the paper. HSV ranging is used to get the area of the paper.
- Input: 

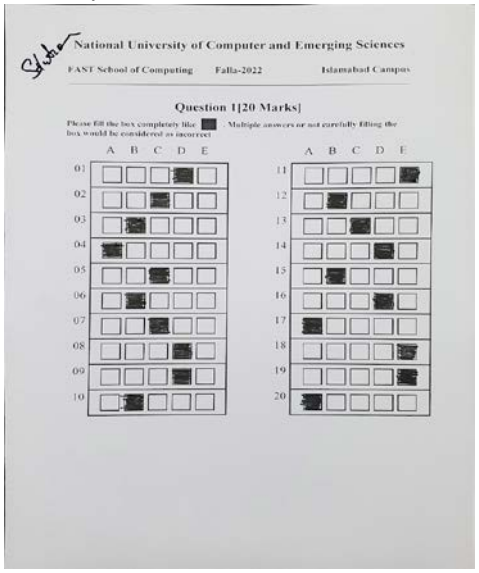
- Output: 

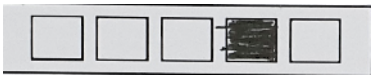
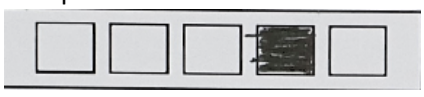
- Canny edge detection is used to get edges of the paper
- Input: 

- Output 

- Contours are then detected of the image to find the boundaries of the paper
  - Input: 

- Output: 

- Warping is done on the contours detected to strech the image properly for better processing. Along with that, the image is resized to 1000,1200 so that each image is processed the same way.
  - Output: 

- The major part of image processing is completed.
- Along with the step above, other contours were also detected i.e. the MCQs. The MCQs are then sorted with respect to their position accordding to the MCQ numbering i.e. 1,2,3 …. 20 in this case.
  - Output: 
- The MCQ boxes are then split into 5 so that each box can be processed individually. Each smaller box is then processed to detect a connected component of saturated black pixels.
  - Input: 

- Output: 
- With each box, their respective option is added into a list.
- The list is then evaluated with a solution that was initially processed in the same way and the score is calculated.