Introduction to Computer Vision and Image Processing

Project 1: Optical Character Recognition

Introduction:

In this project we are asked to build an Optical Character Recognition (OCR) system. OCR means to recognize characters/text from digital/scanned images of printed, handwritten text. It can be used for converting any printed media into electronic form which can be accessible through any computer device. The goal of this project is to detect and recognize various characters using connected component labeling and feature matching algorithms. The steps for the entire operation of the project are described as below. The result is stored in results.json file.

Implementation:

It is a three step process:

1. **Enrollment:** In enrollment , I am extracting features from the images given in the character folder.The character folder contains individual images of letters and numbers. Here we are using the SIFT feature extractor , to get features from the images. While applying SIFT feature extractor we are extracting keypoints and descriptors and storing the descriptors in the feature dictionary , with the character name as its key. I am saving the features in the .json file, so we can get the features for further use.
2. **Detection:** In detection , we are given a test image which has a few sentences with different font size and colors on it. We are required to detect the various characters (letters, numbers, comma, dot) in the image. Here we are using a connected components algorithm to detect various characters in the image and extract each and every character from the image. In the connected component algorithm we are using the DFS method to get the characters from the image. Here I have converted the given test image to grayscale image , and then to binary image. Here I have used thresholding to convert the grayscale image to binary image. Pixels with value <127 will be converted to 1 and the rest of the pixels will be converted to 0.
3. In this image we travers through one pixel one by one in a row. If we encounter a ‘1’ in the binary image , then that pixel coordinate goes to the DFS function. After that , we are checking its neighbor pixels, if the value of the neighbor pixels are 1 then they are appended to a queue. And the parent pixel is appended in the visited array. After that, one of the pixel from the queue is popped and its neighbors are checked, if the value of their neighbor is 1 then its neighbors are pushed into the queue. This is how the entire character is traversed in the image till the queue is completely empty. While traversing the character, we are calculating the minimum and maximum value of the x and y coordinate, to get the starting point of the character and its width and height. In this algorithm I am using the four neighbor method technique , which means we are looking at the neighbor above, below , left and right. Then all the coordinates are stored in an array.

4. **Recognition:** After extracting the bounding box , we are getting each and every character from the image and storing it in the array. Then we are using feature extraction to get the descriptors of the extracted characters and comparing them with the descriptors of the letters which we had extracted earlier and stored in a .json file. To compare the descriptors we are removing Shortest Square Distance. Whoever has the lowest score is given that character is given the name or Unknown written instead of the character's name. All the results are stored in the results.json file.

Result:

The accuracy of the above SIFT model when evaluated over the groundtruth.json is 68.23%