OCR

IDEAZ EC

Pulkit Jain#1, Ravitashaw Bhatla#2

College of Technology  
G.B. Pant University of Agriculture and Technology

Pantnagar

*E-mail:* #1*pulkitjain1004@gmail.com, #2ravitashaw@gmail.com*

*Contact No: +91-8909013992, +91-*

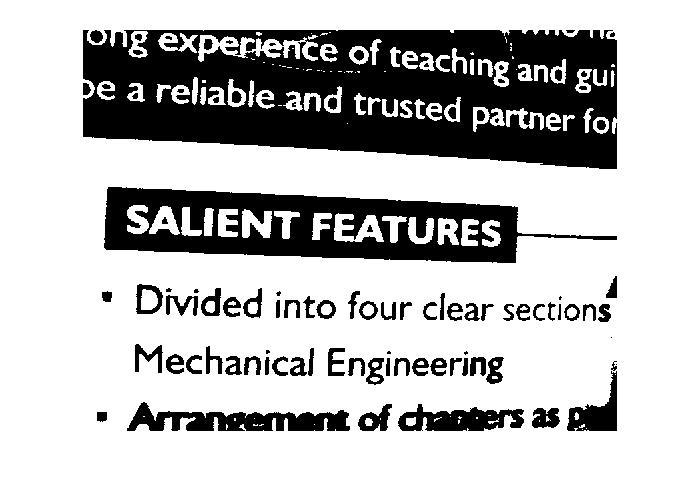
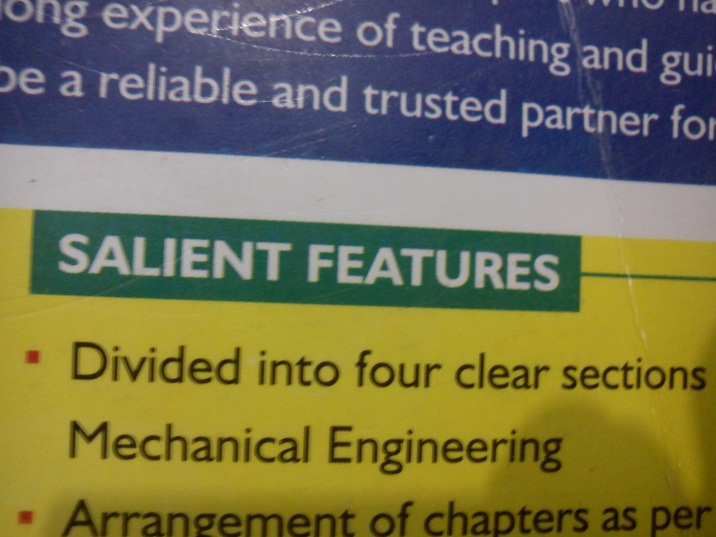
This paper presents algorithms for enhancement of real world photographs for accurate recognition of texts, given an Android based mobile phone under different ambiances. Optical Character Recognition (OCR) from photo is a difficult task, largely due to the several variations in environmental as well as anthropological factors such as uneven illumination, different perspectives and backgrounds. First, the development and execution of the algorithm on MATLAB is discussed, and then its implementation on Android based mobile phone. For character recognition we have used Tesseract, which is an open source OCR engine currently developed by Google.

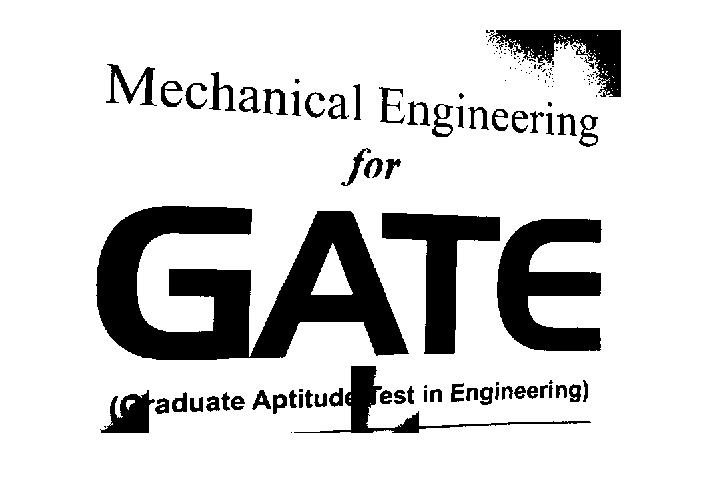
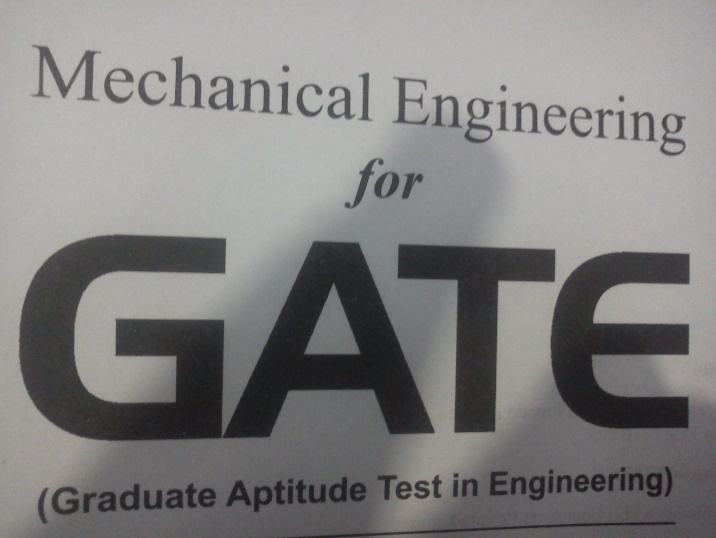
A Photo OCR pipeline involves text detection, character segmentation and character classification. Tesseract, which is considered as the best open source OCR engine and has been used in Google books, works on the following algorithm. It takes a Grayscale or colour image as input and then performs thresholding using OTSU’s method. Then it identifies foreground pixels in the image and marks them as blobs (or potential characters), detects lines of text in the image and determines height and width of all the characters present. And finally, it recognizes character for all the blobs. However, the input data must be from a flat-bed scanner and no rectification capabilities have been provided owing to deviations.

To overcome the above mentioned limitations, we had to transform the input image into a clear binary image. To achieve this, we decided to convert the input image to grayscale to eliminate differences due to colour variations, and also resized it to 800\*600 to make it computationally affordable. Then we segmented the image and performed adaptive thresholding so as to effectively separate characters form everything else. Our results showed that the size of the segments when close to the character dimensions gives best results, thus we bathla the image and segmented it in 50\*50, 100\*100 or 200\*200 w.r.t. character size. Consecutively, we were storing the thresholded segments in a different array. After that we performed noise reduction and morphological operations such as opening, closing and erosion which tidied our image and made characters clear. Finally, we performed Hough transformation and found image boundaries and rescaled it for perspective correction.

The images were then input to the Tesseract OCR and outputs were obtained. Google has already made a wrapper of Tesseract (originally written in C++) for Java under “Eyes Free” project open to public, this helped us in its implementation, since Android in Java based.

The image shown (taken from phone) below has different colours, varying illumination and shadows, disoriented and is slightly blurred and is enhanced using our algorithm.





**References:**

1. Video lec
2. Book
3. S. Bhaskar, N. Lavassar and S. Green, “Implementing Optical Character Recognition on the Android Operating System for Business Cards”.
4. A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, David J. Wu and A.Y. Ng, “Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning”.