**OCR for Camera based handled Device**

Maneesh Kumar Meena

Department of Electrical Engineering

Institute of Technology,Kanpur

Email : [manee@iitk.ac.in](mailto:manee@iitk.ac.in)

Arvind Kumar Meena Department of Electrical Engineering

Institute of Technology,Kanpur

Email : [arvindkm@iitk.ac.in](mailto:arvindkm@iitk.ac.in)

***Abstract***- **The goal of this project is to make an appl-ication for scanning text images to produce an editable output text document with decent quality and with more accurately. OCR and some other techniques are used to perform this operation. Other image sagementa tion algorithmes are used to improe the performace of OCR. After performing all kind of pre-processing step, image is used for ocr operation which extract and separate the textual information from the image and save it to editable document. Some variables can be used to improve the pre-processing step to improve OCR results. Some post-processing steps to perform file operations. The application achived good amount of results on the test images which we collected. We introduced total 5 function in the application, all 5 algorithms performes batter in their region of operations(region- type of background). The results are evaluated for different-different images in the form of TPR(true positive rate). Results are compaired with the normal ocr opeation. Outcomes revealed that this application preformed better almost in all cases. TR**

**1 . INTRODUCTION**

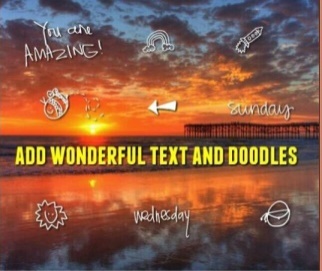
Now a day most of the textual infomation are kept in the form of computer data, use of printed documents is being reduced. Some of the Electronic devices are used for converting these texual information into computer data, example like scanners. Why we keep these data inthe form of images when a simple application can convert into editable document. Specialized scanners are large effective means of converting hard documents into usable soft formats but there are cases where there is need of editiable document instead of scanned document, in those cases this application become quit useful. Some time editable documents has more importance then the information kept as image, in those cases OCR becomes a handy tool. OCR may be needed to make information editable so we can change the information. Further application for ocr is to recognise hand written text, Example- OCR in smart phones. On the other hand, use of camera based hand-held increases which increase the requirment of soft documents. So, we make useof these devices to generate good quality documents.

This project develops a application which help the user scan images with texts. The target is make the output of as good as possible, in quality and accuracy. Some of the image processing techniques are applied in sequences. These processes find out the region of interest and and give its final output which is used to produce final document with editable information, which can be modified further by the user.OCR function is already developed by matlab or anyother platform, but it has limitation. Things become difficult when text image has a complicated backgroud or wirtten text does not satisfy all the properties of ocr. To come over these limitation, we design an app which enhances the image to come over background related difficulties. and we also develope our own OCR which come over the limitation of the previous OCR. We created 3 of our own algorithm and 1 matlab inbuilt algorithm and used different-different kind of textual image to compair these results.

We have used different type of images for our project. Images with simple background , complicated background ,blured images and we compare our results to different algorithms. Here are some Sample Images -

Source : ref. Paper Source – ref. paper

Source : Internet resources source – Internet resources

**2 . METHODOLOGY**

We have used many steps to convert an image to an editable document . First we convert our colored image to gray level value and conduct all operations on it .Here is block diagram of our approach

Gray Scale Image

Text Region Extraction

Skew Correction

Binarization

Text Extraction

Segmentation

Recognisation

Text Document

Fig . Block Diagram of Current Approach

2.1 Text Region Extraction

In this section input gray scale image is segmented into two region – Possible Text Region and Non -text Region. After than we remove Non-Text Region and after that we de-skew Text region,binarize it,Extract text ,do segmentation and recognise text.

Text Region Extraction is done in two processes.

I . Edge Detection

II. Possible Text Region Extraction

I . Edge Detection

We first convert gray scale input image Imxn into edge image . We used a simple method to convert gray scale into edge image. Our used the fact that charecter edge or contour have high contrast than its neighbourhood pixels. In this process edge of text character and some other object that have high contrast are extracted. After that we remove all pixes that have value lower than a threshold value and then sharpen the edge image . Here is pseudo algorithm –

*Input image* ***Imxn*** *ch*

*Output image* ***Iedgemxn***

*Left = 0, Right = 0 , UpperRight = 0*

***for*** *all pixelx,y  in* ***Imxn***

*Left = | pixelx,y - pixelx-1,y  |*

*Right = | pixelx,y - pixelx,y-1  |*

*UpperRight = | pixelx,y – pixelx+1,y-1 |*

***Iedgex,y*** *= max { Left ,Right ,UpperRight } >* ***threshold***

***end***

*sharpen* ***Iedge***

Here threshold is prefixed value to remove background difference.It can be different for different type images.

II . Possible Text Region Extraction

From the computed edge image we extract possible text region based on the fact that a text line candidate contain sufficient number of sharp difference edge pixel. For this first we compute histogram of edge image ***H*** and than ***pri*** – number of pixel in line i greater than a minimum threshold value . similarly we compute ***pcj*** –number of pixel in column j greater than a minimum threshold value . In this way possible text region is extracted . After that we enhance the text tegion image. Here is pseudo region algorithm-

*Input Edge Image* ***Iedgemxn***

*Output image* ***ItextRegionmxn***

***for*** *all* ***pri*** *in* ***H***

***if pri*** *>* ***minThreshold1*** *or (* ***pri – pri-1*** *) >* ***minLinediff***

***ItextRegion*** *( i , :) =****Imxn*** *( i , :);*

***else***

***ItextRegion*** *( i, :) =0;*

***end if***

***If pcj*** *>* ***minThreshold2*** *or (* ***pcj –pcj-1****) >****mincoldiff***

***ItextRegion*** *( :, j) =****Imxn*** *( : , j);*

***else***

***ItextRegion*** *( :, j) =0;*

***end if***

***end for***

Here minThreshold1, minthreshold2,minLinediff,mincoldiff are prefixed value based on image histogram and image type.

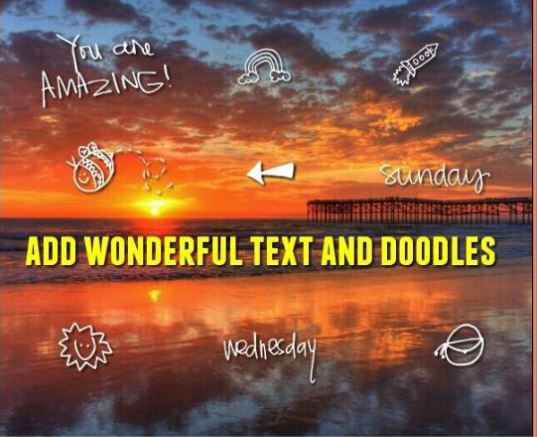


Fig – origina image

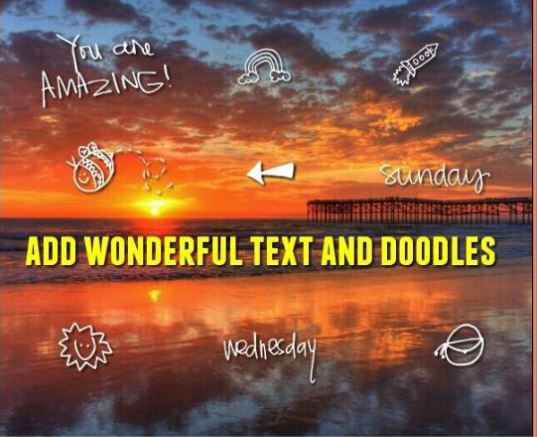


Fig - edge image

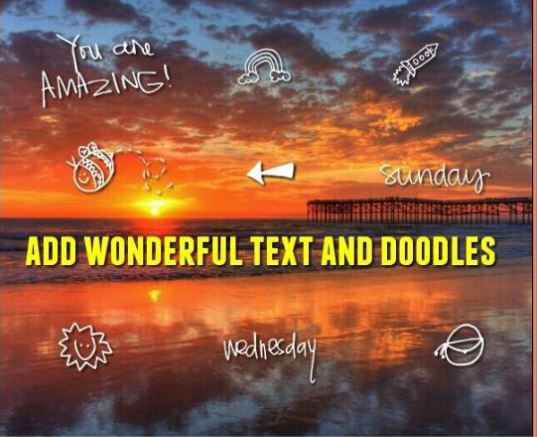


Fig – possible Text Region

2.2 Skew Correction

In an image a skew can occur due to unparallel axis or plane at the time of image capturing . In text region skew can vary between +α to –α where α is a positive number. In Text region Image from thr rectangular boundary of possible text region we calculate profile value . lets M is the length of rectangular value and ***h1***,***h2*** ***h3***....***hi*** are profile value. Then we calculate average ***µ*** and first moment ***β*** and we exclude all profile value that is not within range from ***µ*** + ***β*** to ***µ*** -***β***. After that we calculate skew angle from leftmost ***h1*** , right most ***h***mand middle one ***hc*** and rotate the rectangular by same angle.

**h1 h2 hc  hi-1 hm**

**M**

Average ***µ =(1/M) ∑ hi***

First Moment ***β = (1/M) ∑ | µ- hi |***

Skew angle ***α*** = average ( ***h1-hc  , h1-hm , hm-hc***)

2.3 Binarization

After Skew correction we binarize the image based on a threshold value that gives the best result.Generellay we take average of maximum and minimum gray scale value around a pixel. In binarization we eliminate background around text and other text similar object. Here is pseudo code of binarization.

*Input image* ***ItextRegionmxn***

*Output image* ***Ibinarize***

***for*** *all* ***pixelx,y*** *in* ***ItextRegion***

***if*** *pixelx,y <* ***threshold*** *or neighbourhood around pixelx,y >* ***4***

*pixel****x,y*** *=0*

***end if***

***end for***

2.4 Text Extraction

After Binarize the image we segment our image into text and not text part. Text are segmentated usin geometric property like text width , height and width to height ratio . Text area is also used for removal or non text objects. We discard any other object that fail to satisfy the property of an test . Text are different from any other object. They have about same height in an text line and there area is also fixed and limited. In this process we remove any other object like boundary, any other image and other non text object. The binary edge image is generated from the edge image, erasing all pixels outside the pre defined text boxes and then binarizing it. Then we do gap filling. If one white pixel on the binary edge image is surrounded by two black pixels in horizontal, vertical or diagonal direction, then it is also filled with black. The gap image is used as a reference image to redefine the localization of the detected text candidates. Here is pseudo code of process

*Input image* ***Ibinarizemxn***

*Output image* ***Itextmxn***

***for*** *all object in* ***Ibinarizemxn***

***If*** *(number of pixel in an object <* ***theshold\_value1 )***

*remove that object*

***else if*** *( number of pixel in an object ) >* ***threshold\_value2***

***remove that object***

***else if (*** *width of object**<* ***tw1 )***

***remove that object***

***else if (*** *width of object**>* ***tw2 )***

*Remove that object*

***else if*** *( height of object <* ***th1)***

*remove that object*

***else if (*** *height of object >* ***th2)***

*remove that object*

***else if ( tr1*** *< object area**<* ***tr2)***

*keep object in itext*

***end if***

***end for***

Here ***tw1*** and ***tw2*** are predefined text width limit and ***th1*** and ***th2*** are

predefined text height and ***area*** is limited by maximum and minimum

area of an text. Threshold\_value1 and threshold\_value2 are fixed value and also predefined . They are used to remove non connected non text objects.

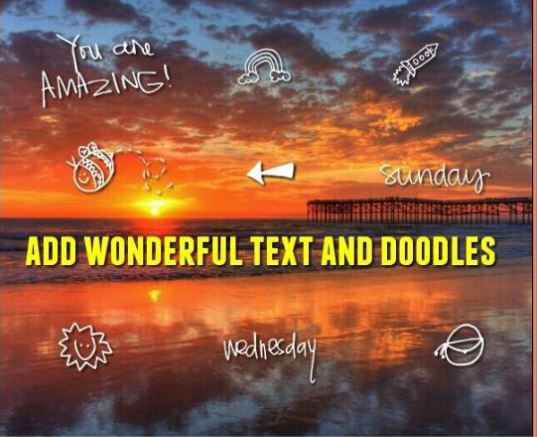


Fig – Binarize image

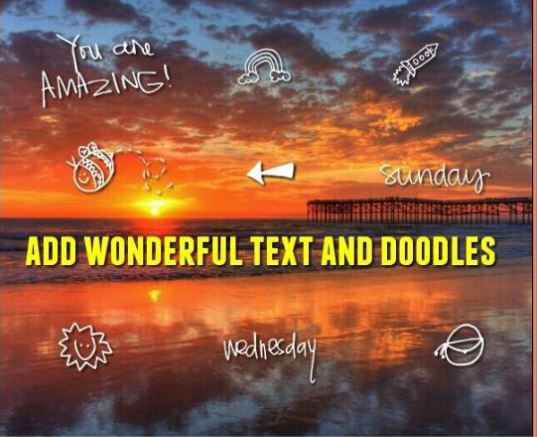


Fig – text Image

2.5 Segmentation

After extraction only possible text object we segment our image into lines and then into individual character. We determine all possible line segments by thresholding the profile values. We choose threshold such that it allow us over-segmentation. Then text line boundaries are choosed by the values of 𝑖 for which the value of 𝑓𝑖 ( number of white pixel in line ) is less than the threshold. If there are n lines there will be n-1 text segmentation . After that we measure distance between segment line and if distance between two segment is too small we reject that segment. After line segmentation we do individual charecter segmentation.

We use vertical histogram profile of each individual text lines to segment individual text .

2.6 Character Recognisation

After doing segmentation we take individual charecter object and using direct ocr function or template matching we recognise the character. If we use template matching there are certain steps . first we take the charecter and select its boundary. After that we normalize its width and height similar to our template charecter. After normalization we match that charecter to every template chatecter and calculate erroe square norm between two charecters. We recognise that charecter that gives us minimmum error square error. Then we write that charecter into our text file .

Binarized image

Boundary Selection

Size Normalization

Template Matching

Charecter Recognisation

Recognised Charecter

Calculation of error norm :

Teemplate is presented by an matrix ***T mxn***and normalized text is represented by ***T***’***mxn*** .then error norm .For all pixel x,y in T

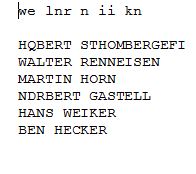
***∑ | t(x,y) –t’(x,y) |2***

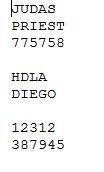
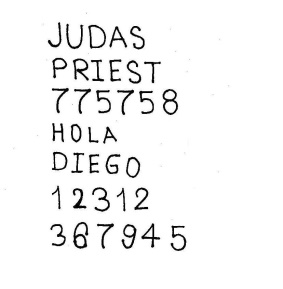
Then we take minimum error norm.

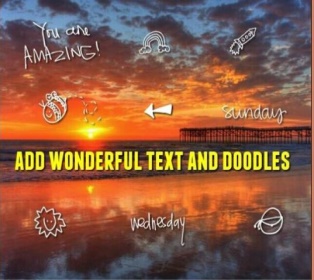
***Expremental Results :***

Using all other processess mention above we carried our experement on sample images.





As based on the result that are carried out on sample images we calculate TPR – true positive rate. Based on Tpr result we see that only a few word are wrongly recognise.It is because either charecter is broken or segmentation is not carried properly for that charecter.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Image Type/ text detectors | OCR | EEH | MSE | MSER |
| Hand Written |  |  |  |  |
| Discontinuous text | 0.5608 | 0.7878 | - |  |
| Background image | 0.0379 | 0.9240 | - |  |
| Blur and complicated | 0.0000 | 0.9342 | - |  |

Above results we conclude that our algorithm shows accuracy above 80%.

**Conclusion**

In this project report we present a complete OCR system .

Because of time complexity we conduct our experiment only on small size images .If we compare to Tesseract, and direct ocr function of matlab our code acquired recognition accuracy (92.74%) . That is good enough. Experiments shows that the recognition system presented in this paper is computationally efficient which makes it applicable for low computing architectures such as mobile phones, personal digital assistants (PDA) etc. If we add different type of images in our template , recognisation can be more efficient. For Hand written text sample images in template should be enough.

**Reference :**

**1 .** Ayatullah Faruk Mollah, Subhadip Basu, and Mita Nasipuri,”Design of an Optical Character Recognition System for Camera-based Handheld Devices “ .( Research Paper from [*www.reasearchgate.net*](http://www.reasearchgate.net/) *)*

2 . Line Eikvil ,”Optical Character Recognition”, December 1993

3. MSER code link

*https://in.mathworks.com/help/vision/examples/automatically-*

*detect-and-recognize-text-in-natural-images.html*