Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study

Article *in* International Journal of Computer Applications · October 2012

DOI: 10.5120/8794-2784

3 authors, including:

Chirag Indravadanbhai Patel
Charotar University of Science and Technology
13 PUBLICATIONS **643** CITATIONS

SEE PROFILE

Dharmendra Patel
Charotar University of Science and Technology
45 PUBLICATIONS **579** CITATIONS

SEE PROFILE

Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study

Chirag Patel
Smt. Chandaben Mohanbhai Patel Institute of Computer Applications (CMPICA)
Charotar University of Science and Technology(CHARUSAT)

Atul Patel, PhD.
Smt. Chandaben Mohanbhai Patel Institute of Computer Applications (CMPICA)
Charotar University of Science and Technology(CHARUSAT)

Dharmendra Patel
Smt. Chandaben Mohanbhai Patel Institute of Computer Applications (CMPICA)
Charotar University of Science and Technology(CHARUSAT)

ABSTRACT
Optical character recognition (OCR) method has been used in converting printed text into editable text. OCR is very useful and popular method in various applications. Accuracy of OCR can be dependent on text preprocessing and segmentation algorithms. Sometimes it is difficult to retrieve text from the image because of different size, style, orientation, complex background of image etc. We begin this paper with an introduction of Optical Character Recognition (OCR) method, History of Open Source OCR tool Tesseract, architecture of it and experiment result of OCR performed by Tesseract on different kinds images are discussed. We conclude this paper by comparative study of this tool with other commercial OCR tool Transym OCR by considering vehicle number plate as input. From vehicle number plate we tried to extract vehicle number by using Tesseract and Transym and compared these tools based on various parameters.

Keywords
Optical Character Recognition (OCR), Open Source, DLL, Tesseract, Transym

1. INTRODUCTION TO OPTICAL CHARACTER RECOGNITION (OCR)
Optical character Recognition (OCR) is a conversion of scanned or printed text images [1], handwritten text into editable text for further processing. This technology allows machine to recognize the text automatically. It is like combination of eye and mind of human body. An eye can view the text from the images but actually the brain processes as well as interprets that extracted text read by eye. In development of computerized OCR system, few problems can occur. First: there is very little visible difference between some letters and digits for computers to understand. For example it might be difficult for the computer to differentiate between digit "0" and letter "o". Second: It might be very difficult to extract text, which is embedded in very dark background or printed on other words or graphics. In 1955, the first commercial system was installed at the reader's digest, which used OCR to input sales report into a computer and then after OCR method has become very helpful in computerizing the physical office documents. There are many applications of OCR, which includes: License plate recognition [2], [3], [4], [5], [13],[14],[16],[17],[19], image text extraction from natural scene images [6], extracting text from scanned documents [12] etc. The system proposed in [12 is to rectify the text retrieved from camera captured images. An OCR system proposed by Thomas Deselaers *et al.* [15] is used for recognizing handwritten characters and converting these characters into digital text. A system presented by Apurva A. Desai [18] is used to recognize Guajarati handwritten numeral by using Artificial Neural Network (ANN). This system is able to achieve average 82% recognition of Gujarati digits. U. Pal *et al*. [20] used Support Vector Machines (SVM) method for Bangla and Devnagari text recognition. This system is able to achieve 99.18% accuracy for Devnagari and 98.86% accuracy for Bangla characters. Bilal *et al.* [21] suggested local binarization method for document images by using a Thresholding method and dynamic windows. This system is able to achieve F-mean values of 85.1% for handwritten text and 90.93% for printed text. A survey paper by Fink et al. [22], describes various OCR systems based on the Markov models. Various steps regarding OCR are explained in details. In this paper details regarding markov model for handwritten character are explained.

Today many types of OCR software available in the markets like: Desktop OCR, Server OCR, Web OCR etc. Accuracy rate of any OCR tool varies from 71% to 98%. Many OCR tools are available as of now but only few of them are open source and free. Tesseract is the one of the open source and free OCR software [7]. It is written in the C++, so it is platform independent. It can be used in other applications in the form of Dynamic Link Library (DLL). So it can be easily added as the reference in the form of DLL in other application to use the functionality provided by Tesseract. The following section contains history and functionality of Tesseract.

2. HISTORY OF TESSERACT
Tesseract is an open source optical character recognition engine [7]. It was developed at HP in between 1984 to1994 [7]. It was modified and improved in 1995 with greater accuracy. In late 2005, HP released Tesseract for open source. It is now available at [8]. It is highly portable. It is more focused towards providing less rejection than accuracy. Currently only command base version is available. As of now Tesseract version 3.01 is released and available for use. HP never used it. Now it is developed and maintained by Google. It provides support for various languages.

3. ARCHITECTURE OF TESSERACT
Tesseract OCR works in step by step manner as per the block diagram shown in fig. 1. First step is Adaptive Thresholding [9], which converts the image into binary images. Next step is connected component analysis [7], which is used to extract

character outlines. This method is very useful because it does the OCR of image with white text and black background.

Tesseract was probably first [7] to provide this kind of processing. Then after, the outlines are converted into Blobs. Blobs are organized into text lines, and the lines and regions are analyzed for some fixed area or equivalent text size [7]. Text is divided into words using definite spaces and fuzzy spaces [7]. Recognition of text is then started as two-pass

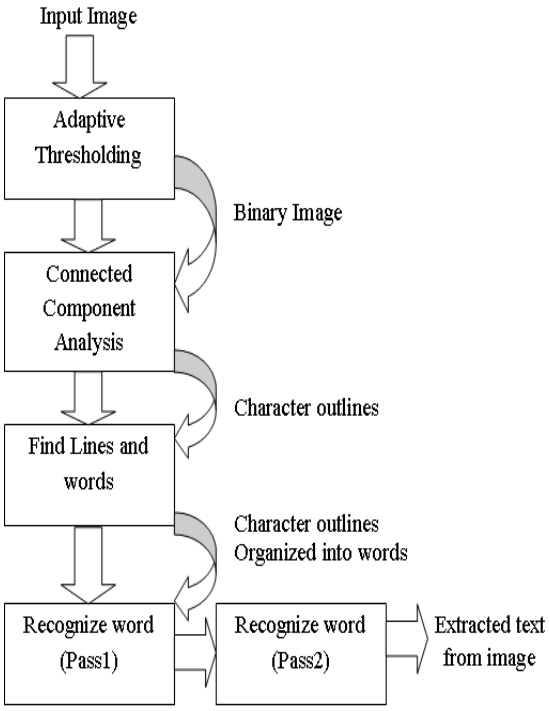

Fig. 1. Architecture of Tesseract OCR

process as shown in fig 1. In the first pass, an attempt is made to recognize each word from the text. Each word passed satisfactory is passed to an adaptive classifier as training data [7]. The adaptive classifier tries to recognize text in more accurate manner. As adaptive classifier has received some training data it has learn something new so final phase is used to resolve various issues and to extract text from images. More details regarding every phase are available at [7].

4. Working of Tesseract

An image with the text is given as input to the Tesseract engine that is command based tool. The image is shown in fig.2 [10]. Then it is processed by Tesseract command as shown in fig.3. Tesseract command takes two arguments: First argument is image file name that contains text and second argument is output text file in which, extracted text is stored. The output file extension is given as .txt by Tesseract, so no need to specify the file extension while specifying the output

Fig. 2. Image containing text

file name as a second argument in Tesseract command.

As Tesseract supports various languages, the language training data file must be kept in the tessdata folder. In this

Fig. 3. Processing of image in Fig 2 by Tesseract OCR

research, the purpose is to extract English text from the images so we have kept only English language file in the tessdata folder. After processing is completed, the content of the output file shown in fig 4. In simple images with or without color (gray scale), Tesseract provides results with 100% accuracy. But in the case of some complex images Tesseract provides better accuracy results if the images are in the gray scale mode as compared to color images. To prove this hypothesis, OCR of same color images and gray scale images is performed and in both cases different result are achieved.

Fig. 4. Output of processing done by Tesseract on Fig. 2

4.1 OCR of color image by Tesseract

OCR of a complex color image shown in the fig. 5 [11] is performed by Tesseract and after OCR processing of image; the text extracted in the image is not as accurate as it is expected. As it is visible in the fig 6, the extracted text is not exactly same as it is visible in the image of fig 5. That means Tesseract is not able to extract the text with 100% accuracy. So to get more accurate result the same image is converted to the gray scale image and OCR is performed on this image.

Fig. 5. Color Image

The experiment result is discussed in the following section.

4.2 OCR of gray scale image by Tesseract

As it is visible in the fig 7, the color image of fig. 5 is converted in the gray scale image. The image is converted to the gray scale by using the algorithm that is discussed in the following section. After processing the above image output is

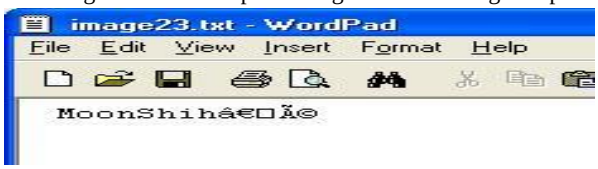

Fig. 6. Output of OCR processing of Fig.5 by Tesseract

shown in the fig 8. It is clearly visible that Tesseract has successfully retrieved the text "MoonShine", which is expected as output. This means that Tesseract has extracted the text with 100% accuracy. The results discussed in section

Fig. 7. Gray Scale Image

4.4 reveal that Tesseract provides more accuracy in processing gray scale images. The following section contains algorithm to convert any color image to the gray scale image.

4.3 Algorithm to convert color image to gray scale image

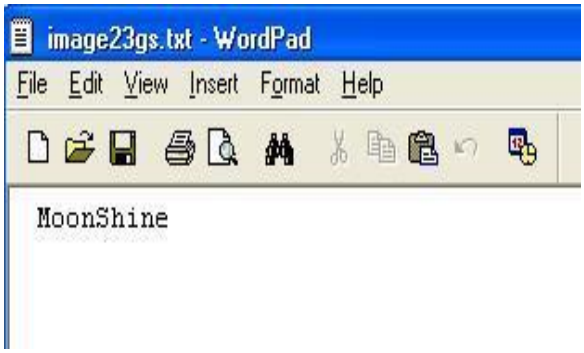

Fig. 8. Output of OCR processing done by Tesseract on image of Fig 7

A digital image with M width (row) and N height (column) is represented as discrete function f(x, y) as:

$$f(x,y) = (xi, yj), \text{where i} = 0, \text{i} < N, j = 0, j < M \quad (1)$$

Here the pair (xi, yj) is known as pixel. The pair (0,0) is the first pixel and pair(M-1,N-1) is the last pixel in the image. Every pixel has its own RGB color value. If the pixel has the same RGB value then it falls into gray color family (black to white). So based on this observation the algorithm to convert color image to gray scale is developed, which is shown under:

$$\mu(x,y) = \frac{\Sigma\big((x,y)r, (x,y)g, (x,y)b\big)}{3} \quad \forall(x,y)\ where$$

$$-1 < r, g, b < 256 \quad (2)$$

Here r, g and b are red color, green color, blue color values of pixel (x, y) respectively. Range of r, g and b are mentioned in (2) and μ is the mean value of these pixels that is always less than 256. So value of μ is in between 0 to 255, which is assigned to red, green and blue pixels of pixel (x, y). This process is depicted in (3), (4) and (5).

$$(\text{x,y})\text{r} = \mu(x,y)\ \forall(x,y)\ where\ x \in N, y \in M \quad (3)$$

$$(\text{x,y})\text{g} = \mu(x,y) \forall(x,y)\ where\ x \in N, y \in M \quad (4)$$

$$(\text{x,y})\text{b} = \mu(x,y) \forall(x,y)\ where\ x \in N, y \in M \quad (5)$$

So after applying above algorithm on color image it becomes gray scale image.

4.4 Experiment results of OCR by Tesseract

We have captured 20 different kinds of number plates' images from various types of vehicles and performed OCR of these number plates to extract vehicle number. Table I shows that Tesseract provides 61% accuracy with the color images and 70% accuracy with gray scale images. So it indicates that Tesseract provides better accuracy in gray scale images as compared to color images. This experiment was carried out on computer with Intel Pentium (R) 4 2.4GHZ CPU and 1 GB RAM. The images of number plates are captured by 5-mega-pixel camera. We can observe that if color images are converted to gray scale and given as input to Tesseract then accuracy of text extraction is increased. In some color images where text extraction accuracy result is 100 % or near to 100 %, and if it is converted to gray scale then it produces same amount of extraction accuracy. In some color images Tesseract is not able to provide more than 40 % of accuracy; we have converted these images into gray scale images by using the algorithm discussed in the previous section and then given these images as input to Tesseract. So after doing this process there is an increase in the average accuracy to extract the characters from the vehicle number plate. This accuracy of individual image processing varies from 16% to 100%, which can be seen in Table I with the image numbers 10 to 18. Also it is observed that the processing time of extracting characters from gray scale images is decreased. It is reduced by 10% to 50%. So we can say that Tesseract works fast and provides better text extraction accuracy in processing gray scale number plate images.

TABLE I

Tesseract OCR Result analysis

Image No	Image Type	Number of character in image	No of characters extracted	Accuracy of OCR of color images (in Percentages)	Time taken for OCR (in Seconds)	Image Type	No of characters extracted after converting color to gray scale image	Accuracy of OCR of gray scale images (in Percentages)	Time taken for OCR (in Seconds)	Change in Accuracy (In percentages)
1	color	**12**	5	42	0.4	gray scale	5	42	0.397	
2	color	**12**	12	100	0.202	color	12	100	0.202	
3	color	**12**	8	67	0.301	gray scale	8	67	0.601	
4	color	**9**	9	100	0.5	color	9	100	0.5	
5	color	**8**	8	100	0.505	color	8	100	0.505	
6	color	**9**	7	78	0.909	gray scale	7	78	0.909	
7	color	**8**	8	100	0.805	color	8	100	0.805	
8	color	**9**	7	78	1.01	gray scale	7	78	1.01	
9	color	**10**	7	70	0.85	gray scale	7	70	0.798	
10	color	**9**	4	44	0.907	gray scale	5	56	0.402	20
11	color	**10**	1	10	1.007	gray scale	4	40	0.548	75
12	color	**10**	4	40	0.699	gray scale	7	70	0.402	42.86
13	color	**10**	3	30	1.51	gray scale	4	40	0.701	25
14	color	**9**	0	0	1.008	gray scale	4	44	0.705	100
15	color	**9**	0	0	1.815	gray scale	2	22	0.7	100
16	color	**11**	6	55	1.619	gray scale	8	73	1.717	25
17	color	**9**	5	56	0.99	gray scale	6	67	0.806	16.67
18	color	**11**	5	45	0.907	gray scale	6	55	0.596	16.67
19	color	**9**	9	100	3.048	color	9	100	3.048	
20	color	**9**	9	100	1.007	color	9	100	1.007	
			Average Accuracy	**61**			**Average Accuracy**	**70**		

5. Comparison Study of Tesseract OCR with Transym

Transym OCR is one of the proprietary Optical Character Recognition tools, which also provides the good amount of accuracy, so we tried to perform OCR on the same set of images, which are discussed in section 4.4 to observe the result produced by Transym. Transym coverts the color images to gray scale images then does the OCR of these images. So we do not need to convert color image to gray scale while using Transym. The summary of OCR processing results by Transym OCR is shown in the form of Table II, which shows that Transym provide about 47% of average accuracy in our set of data.

A comparison study of Tesseract is carried out with Transym. Many other OCR tools are proprietary and not open source. Some tools are not able to provide same kind of accuracy that Tesseract provides. As Transym provides great amount of accuracy, we have used trial version of it for doing OCR of images shown as per the table II. It cannot be used in the form of Dynamic Link Library (DLL) but Tesseract can be used in another application in the form of DLL for various uses as it is not a complete tool but it is OCR engine.

From Table III we can say that Tesseract provides better accuracy of 61% for color image and 70% for gray scale images as compared to Transym, which provides only 47% of accuracy in our set of data. Tesseract is faster than Transym because it takes average 1 second and 0.82 seconds for processing color and gray scale images respectively to process one image, while Transym takes average 6.75 seconds to process one image. Standard deviations for accuracy provided by Tesseract for color and gray scale images are 34.21 and 24.64 respectively are also less than the

TABLE II
Transym OCR Result analysis

Image No	Image Type	Number of characters in image	No of characters extracted	Accuracy of OCR (in Percentages)	Time taken for OCR (in Seconds)
1	color	**12**	0	0	54.44
2	color	**12**	10	83	2.115
3	color	**12**	0	0	10.58
4	color	**9**	0	0	2.01
5	color	**8**	8	100	1.466
6	color	**9**	1	11	10.725
7	color	**8**	0	0	7.591
8	color	**9**	6	67	4.589
9	color	**10**	6	60	1.816
10	color	**9**	8	89	2.416
11	color	**10**	0	0	3.264
12	color	**10**	10	100	1.104
13	color	**10**	0	0	1.7
14	color	**9**	7	78	3.827
15	color	**9**	6	67	2.904
16	color	**11**	0	0	2.717
17	color	**9**	5	56	0.502
18	color	**11**	6	55	2.504
19	color	**9**	6	67	17.256
20	color	**9**	9	100	1.513
			Average Accuracy	**47**	

TABLE III
Tesseract and Transym OCR Comparison

Feature	**Tesseract OCR**	**Transym OCR**
Free	Yes	No (Trial Version is available)
Open Source	Yes	No
License	Apache	Proprietary
Online	No	No
Operating System	Window, MaC, Linux	Windows
Latest Stable version	3.01	3
Release Year	2010	2008
DLL Available	Yes	No
Accuracy (For extracting character from vehicle number plate)	61% (color images) 70% (gray scale images)	47%
Average Time	1 second (color images) 0.82 Seconds gray scale images)	6.75 seconds
σ_{AC}	34.21(For color Images) 24.64(For Gray Scale Images)	40
σ_{T}	0.63 (For color images) 0.61(For gray scale images)	12

σ_{AC} = Standard deviation of Accuracy

σ_{T} = Standard deviation of Time taken to perform OCR

Standard deviation of accuracy provided by Transym that is 40. So in this case of providing accuracy of above set of data Tesseract is more consistent. Also the standard deviations of processing time of Tesseract for color images and grays scale images are 0.63 and 0.61 respectively, which are far less than the standard deviation of processing time of Transym that is 12. So in this case of processing images in less time Tesseract is more consistent.

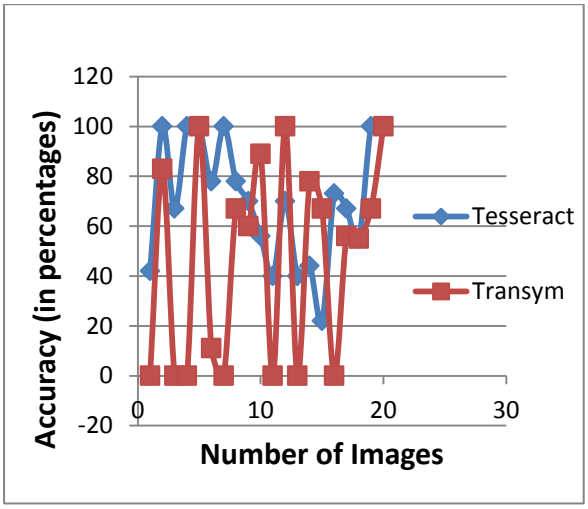

Fig. 9. Comparison of accuracy.

A comparison graph of Tesseract and Transym is shown in fig 9, to study the performance of both tools by considering parameter of accuracy. As we can see in fig 9, for some images Transym provides 0% or near to 0% accuracy i.e. it is not able to extract any character from the those images. While for none of the images Tesseract provides 0% accuracy i.e. it is able to extract at least some characters from every image. This comparison is done after color images are converted to gray scale for Tesseract.

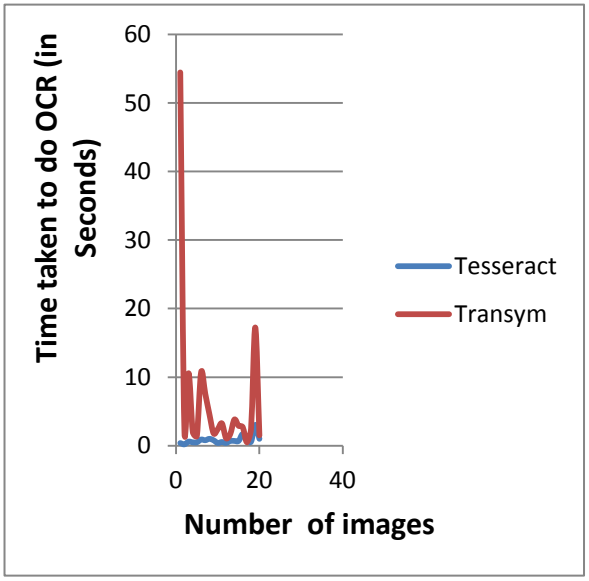

Fig. 10. Comparison of processing time of OCR

Another Comparison based on the processing time of OCR is done in between these two tools. As we can see in graph of fig 10, for many images Tesseract takes hardly 1 second or less for OCR, while it is rare for Transym to take 1 second or less for OCR of any image. For some images it takes around 1 to 18 seconds for OCR processing. As it is visible in fig.10 for one of the images it has taken more than 50 seconds. So we can observe that Tesseract is faster than Transym in OCR processing.

6. CONCLUSION

Although Tesseract is command-based tool but as it is open source and it is available in the form of Dynamic Link Library, it can be easily made available in graphics mode. The results obtained in above sections are obtained by extracting vehicle number from vehicle number plate. So above results do not confirm that Tesseract is always better or faster than Transym but is it more accurate in extracting text from the vehicle number plate. The input images are specific, which are vehicle number plates, so in these specific images Tesseract provides better accuracy and in other kinds on images Transym might provide better accuracy than Tesseract. As we are interested in extracting vehicle number from vehicle number plate, we have considered both tools for serving this specific purpose.

7. ACKNOWLEDGEMENTS

The authors thank MCA Department of Charotar University of Science and Technology (CHARUSAT) for providing required resources to accomplish this research.

8. REFERENCES

[1] ARCHANA A. SHINDE, D**.** 2012.Text Pre-processing and Text Segmentation for OCR. International Journal of Computer Science Engineering and Technology, pp. 810-812.

[2] ANAGNOSTOPOULOS,C.,ANAGNOSTOPOULOS, I., LOUMOS, V, & KAYAFAS, E. 2006. A License Plate Recognition Algorithm for Intelligent Transportation System Applications**.,** IEEE Transactions on Intelligent Transportation Systems, pp. 377- 399.

[3] Y. WEN, Y. L. 2011. An Algorithm for License Plate Recognition Applied to Intelligent Transportation System., IEEE Transactions on Intelligent Systems, pp. 1-16.

[4] XIN FAN, G. L. 2009. Graphical Models for Joint Segmentation and Recognition of License Plate Characters. IEEE Signal Processing Letters, pp. 10-13.

[5] HUI WU, B. L. 2011. License Plate Recognition system. International Conference on Multimedia Technology (ICMT). pp. 5425 - 5427.

[6] PAN, Y.-F., HOU, X., & LIU, C.-L. 2008.A Robust System to Detect and Localize Texts in Natural Scene Images.The Eighth IAPR International Workshop on Document Analysis Systems.

[7] SMITH, R. 2007. An Overview of the Tesseract OCR Engine. In proceedings of Document analysis and

Recognition.. ICDAR 2007. IEEE Ninth International Conference.

[8] GOOGLE. Google Code. google code. [Online] 2012. http://code.google.com/p/tesseract-ocr/.

[9] F. SHAFAIT, D. K. San Jose, CA : s.n., 2008.Efficient Implementation of Local Adaptive Thresholding Techniques Using Integral Images.. In Document Recognition and Retrieval XV, S&T/SPIE Annual Symposium on Electronic Imaging.

[10] 1stwebdesigner. 1stwebdesigner. [Online] 2012. http://www.1stwebdesigner.com/wp-content/uploads/2009/11/typography-tutorial/text1-how-to-create-typographic-wallpaper.jpg.

[11] dsigninspire. Desing Inspire. [Online] 2012. http://dsigninspire.com/wpcontent/uploads/2011/09/moon-shine.jpg.

[12] Geometric Rectification of Camera-Captured Document Images. Jian Liang; DeMenthon, D.; Doermann, D.; April 2008. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.30, no.4, pp.591-605.

[13] Y. WEN, Y. L. 2011.,An Algorithm for License Plate Recognition Applied to Intelligent Transportation System. IEEE Transactions on

Intelligent Systems, pp. 1-16.

[14] Lihong Zheng, Xiangjian He, Bijan Samali, Laurence T. Yang. An algorithm for accuracy enhancement of license plate recognition. Journal of Computer and System Sciences, Available online 9 May 2012.

[15] Deselaers, T.; Gass, T.; Heigold, G.; Ney, H.; Latent Log-Linear Models for Handwritten Digit Classification. June 2012. IEEE Transactions on Pattern Analysis and Machine Intelligence, , vol.34, no.6, pp.1105-1117, doi: 10.1109/TPAMI.2011.218.

[16] Jianbin Jiao, Qixiang Ye, Qingming Huang, A configurable method for multi-style license plate recognition. 2009. Pattern Recognition, Volume 42, Issue 3, , Pages 358-369.

[17]H. Erdinc Kocer, K. Kursat Cevik. 2011.Artificial neural networks based vehicle license plate recognition. Procedia Computer Science, Volume

3, Pages 1033-1037.

[18] Apurva A. Desai. 2010. Gujarati handwritten numeral optical character reorganization through neural network, Pattern Recognition, Volume 43, Issue 7 Pages 2582-2589, ISSN 0031-3203, 10.1016/j.patcog.2010.01.008.

[19] Roy, A.; Ghoshal, D.P. 2011. Number Plate Recognition for use in different countries using an improved segmentation, 2nd National Conference on Emerging Trends and Applications in Computer Science (NCETACS),vol., no., pp.1-5, 4-5. doi: .1109/NCETACS.2011.5751407.

[20] Umapada Pal, Partha Pratim Roy, Nilamadhaba Tripathy, Josep Lladós. December 2010. Multi-oriented Bangla and Devnagari text recognition, Pattern Recognition, Volume 43, Issue 12, Pages 4124-4136, 10.1016/j.patcog.2010.06.017.

[21] Bilal Bataineh, Siti Norul Huda Sheikh Abdullah, Khairuddin Omar. 2011.An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows, Pattern Recognition Letters, Volume 32, Issue 14, , Pages 1805-1813, ISSN 0167-8655, 10.1016/j.patrec.2011.08.001.

[22] Fink, Gernot.2009. Markov models for offline handwriting recognition: a survey. International Journal on Document Analysis and Recognition.Springer Berlin / Heidelberg pp. 269-298,volume: 12,Doi: 10.1007/s10032-009-0098-4

9. AUTHORS PROFILE

Chirag I Patel received Bachelors degree in computer application (B.C.A) from Dharmsinh Desai University Nadiad, Gujarat, India in 2002 and Masters Degree in Computer Applications (M.C.A) from Gujarat University, Gujarat, India in 2005. He is pursuing PhD in Computer Science and Application from Charotar University of Science and Technology (CHARUSAT). He with MCA Department at Smt Chandaben Mohanbhai Patel Institute of Computer Applications, Charotar University of Science and Technology (CHARUSAT), Changa, Gujarat, India. His research interests include Information Retrieval from image/video, Image Processing and Service Oriented Architecture.

Dr. Atul Patel received Bachelors degree in Science (Electronics), M.C.A. Degree from Gujarat University, India. M.Phil. (Computer Science) Degree from Madurai Kamraj University, India. Now he is an Associate Professor and Head, Smt Chandaben Mohanbhai Patel Institute of Computer Applications – Changa, India. He has received Ph.D degree in wireless ad-hoc network from Sardar Patel University. His main research areas are wireless communication and Network Security.

Dharmendra Patel received Bachelors degree in Industrial Chemistry - BSc (Industrial Chemistry) from North Gujarat University, Gujarat, India and Masters Degree in Computer Applications (M.C.A) from North Gujarat University, Gujarat, India. He has pursuing PhD in the field of web usage mining. He is with MCA Department at Smt Chandaben Mohanbhai Patel Institute of Computer Applications, Charotar University of Science and Technology (CHARUSAT), Changa, Gujarat, India. His research interests include Web usage mining.