Efficient multiscale and multifont optical character recognition system based on robust feature description

Mahmoud Soua, Rostom Kachouri, Mohamed Akil

Efficient Multiscale and Multifont Optical Character Recognition System based on robust Feature Description

Mahmoud Soua, Rostom kachouri and Mohamed Akil
Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge, Equipe A3SI, ESIEE Paris, France
e-mail: mahmoud.soua@esiee.fr, rostom.kachouri@esiee.fr, mohamed.akil@esiee.fr

***Abstract*— Optical Character Recognition (OCR) is the process of translating images of text into a comprehensible machine format. Generally, an OCR system is composed of binarization, segmentation and recognition stages. Given an extracted binary character, the recognition stage ensures its description and decides its corresponding ASCII code. In this paper, we propose a new OCR system that aims to high speed, Multiscale and Multifont character recognition. Our proposal is based essentially on robust description using a new Unified Character Descriptor (UCD). In addition, a character type-face and font-size recognition is performed to choose the adequate template for faster matching process. Obtained OCR Accuracy of our proposed System is 1.5x higher then that reached by Tesseract on the LRDE dataset.**

***Keywords*— OCR System, Multiscale, Multifont, Feature Extraction, Feature Matching, SAD technique.**

I. Introduction

Optical Character Recognition (OCR) deals with the problem of recognizing optically processed characters. OCR systems involve three major stages to completely recognize characters; Binarization, Segmentation, and Recognition stages [1]. Firstly, Binarization separates the text characters from the background [6]. Then, Segmentation stage aims to locate text regions in the processed documents [7]. Finally, the Recognition stage consists on a very sensitive character description and decision steps.

Generally, state-of-the-art methods in the description step are based either on Template Description (TD) or on Feature Description (FD) [1]. In TD Methods [1][2], characters are described based on their pixel information. Despite their simple use, these methods are not robust on noisy characters. In addition, calculation are burden with unnecessary pixel description. Less complex, the FD methods [3][4][5] perform the description of characters based on some specific Features.

In the other hand, the decision stage is made according to a matrix or extracted feature matching. As mentioned above, pixel comparison in matrix matching is extremely sensitive to noisy characters [1][2]. This stage can be performed using classification methods such as RNA, SVM [3][4][5]. They give interesting results, however they still complex comparing to matching process. Indeed, simple feature matching ensures a good trade off between accuracy and computation complexity [5].

In this paper, we propose a novel Multiscale and Multifont OCR System based on a robust feature description. Firstly, we ensure the right template selection. Then, we compute a Unified Character Descriptor (UCD) and a fast Matching process is performed.

In the following, we present our proposed OCR System in section 2 where the new template selection and Character Recognition based on UCD (CR-UCD) method is explained. The obtained results are shown and discussed in section 3. Finally conclusion and future work are drawn in section 4.

II. Proposed OCR System based on CR-UCD Recognition

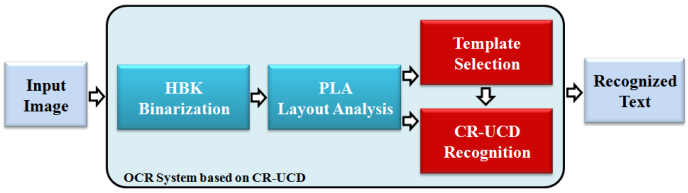


Fig. 1: Block diagram of our proposed OCR System

We design a complete OCR System that performs character recognition on input image containing text information. As seen in Figure 1, our OCR System includes the stages of binarization, segmentation, template selection and recognition. In the binarization stage we use our recently proposed Hybrid Binariaztion Based on Kmeans (HBK) [6][13] method to separate correctly the text and the background even on noisy images. Then, binarized image is segmented with the Page Layout Analysis (PLA) [10] part of the well known Tesseract 3.02 engine [20]. In the template selection stage, character font-size and type-face are recognized and subsequently the adequate template is chosen. Finally, we perform our CR-UCD recognition in which each character is represented firstly in the Description stage using one UCD feature. Then, characters are recognized based on feature matching with the selected template using the Sum of Absolute Difference (SAD). An example of letter '*A*' recognition with our proposed method is given in Figure 2, we assume that the considered letter '*A*' is extracted from the word '*Académies*'. In this figure, the different proposed stages of our new method, namely: the Template Selection, Description and Decision stage are illustrated. In the first stage a Template Selection Descriptor (TSD) is generated to allow the appropriate template choice. In an other hand, the description stage extracts the Unified Character Descriptor (UCD) from each character and feeds it into the decision stage. In this stage the extracted UCD vector is matched with the

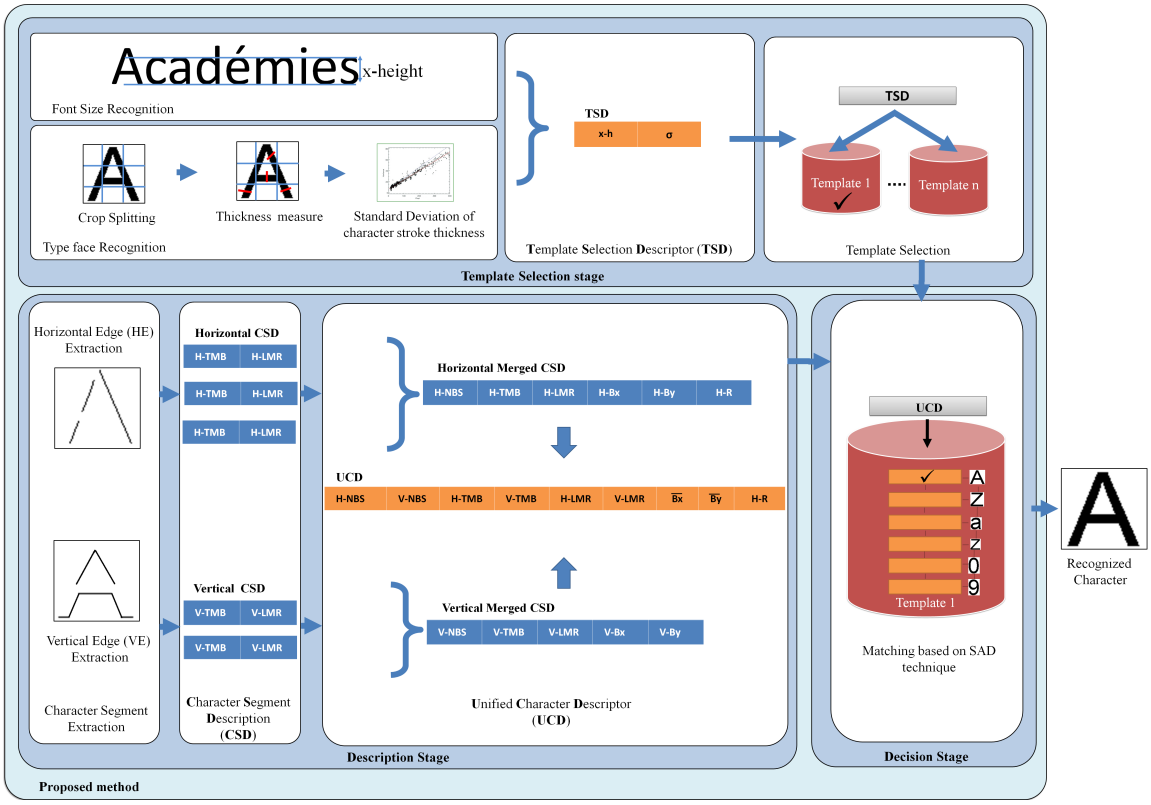


Fig. 2: Letter '*A*' recognition with the proposed method

closest character in the selected template. These stages are given with more details in subsequent sections. Firstly, we explain the Description stage. Then, we discuss the template selection and we present the decision ones.

A. The Description Stage

In this section, we present our Feature Extraction Strategy. The aim of this stage is to employ a sufficient number of characteristics that helps to discriminate characters efficiently. For this, we perform in our work a Character Segment Extraction by using a simple Edge Detection. We extract then horizontal and vertical character segments. In the Horizontal Edge (HE) extraction, for each pixel P_{ij} we compute the right edge according to Equation (1). Where $i \in [1..h]$ and $j \in [1..w]$, given that h and w are respectively the height and width of the character bounding box. We note by 0 a black pixel and by 1 a white one.

$$P_{i,j} \in \begin{cases} \text{HE If } (P_{i,j} = 0) \text{ And } (P_{i,j+1} = 1) \\ \\ \overline{\text{HE}} \text{ If } ((P_{i,j} = 0 \text{ And } (P_{i,j+1} = 0) \text{ Or } (P_{i,j} = 1)) \end{cases} \quad (1)$$

In the other side, the Vertical Edge (VE) extraction, for each pixel P_{ij} is computed as shown in Equation (2).

$$P_{i,j} \in \begin{cases} \text{VE If } (P_{i,j} = 0) \text{ And } (P_{i+1,j} = 1) \\ \\ \overline{\text{VE}} \text{ If } ((P_{i,j} = 0 \text{ And } (P_{i+1,j} = 0) \text{ Or } (P_{i,j} = 1)) \end{cases} \quad (2)$$

Generally, when we re-scale characters or when we handle the same character font in many sizes, we change the character morphology and some distortions can appear in the character description. This issue is named the aliasing behavior [10]. As shown in Figure 3, due to this phenomena, one character can have a different number of segments on multiple Font-Sizes.

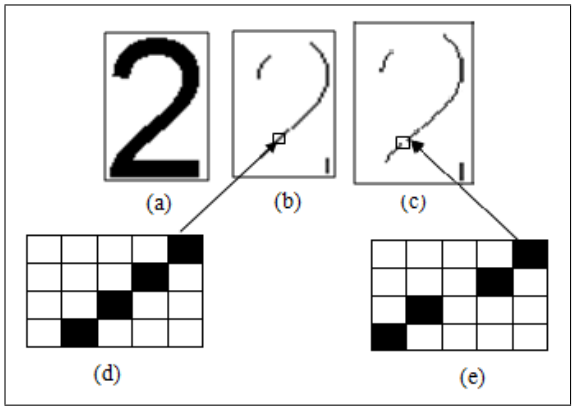


Fig. 3: Dependency of scale and character segment discontinuity: a. Input Images, b. Extracted Horizontal Edges in one scale, c. Extracted Horizontal Edges in another scale, d and e. A corresponding zoom views of the extracted segments

For this, we propose to check the extracted segments to merge some edges based on their neighborhood. To ensure an efficient and scale invariant method, we use each time a

variable neighbour pixel number for the check. This parameter is adjusted according to h and w values of the processed character (its size). This proposed technique allows to improve the description of multi-scale character.

As shown in Figure 2, the Character Segment Description (CSD) of each Horizontal and Vertical extracted segments is firstly ensured by the Character Segment Position (TMB and LMR). Then one Horizontal Merged CSD is constructed while adding respectively H-TMB and H-LMR of each Horizontal segment. To which we concatenate other features like the total Horizontal Character Segment Number (H-NBS), the Horizontal Character Barycentre Coordinates (H-Bx and H-By), and the Horizontal Character Ratio (H-R). We do the same thing for the Vertical Merged CSD computation except the Character Ratio (R) feature which is not used in this case. The employed features in this work are presented and explained with more details in the following subsections:

1) Character segment Number Feature (NBS): We consider that this feature increases the discrimination between characters that do not have the same number of segments.

2) Character Segment Position Feature (TMB, LMR): As shown in Figure 2, the CSD vector is composed of two features: The first one, TMB, refers to the Top, Middle and Bottom segment positions in the character bounding box. We compute TMB according to Equation (3).

$$TMB = \begin{cases} 1 \text{ If } (S_y > \frac{h}{2}) \text{ (Top)} \\ 2 \text{ If } (S_y = \frac{h}{2}) \text{ (Middle)} \\ 3 \text{ If } (S_y < \frac{h}{2}) \text{ (Bottom)} \end{cases} \quad (3)$$

with S_y is the starting y coordinates of the profiled segment and h is the height of the character bounding box. The second feature is LMR. It refers to the Left, Middle and Right segment positions in the character bounding box. It is computed as shown in Equation (4).

$$LMR = \begin{cases} 1 \text{ If } (S_x < \frac{w}{2}) \text{ (Left)} \\ 2 \text{ If } (S_x = \frac{w}{2}) \text{ (Middle)} \\ 3 \text{ If } (S_x > \frac{w}{2}) \text{ (Right)} \end{cases} \quad (4)$$

with S_x is the starting x coordinates of the profiled segment and w is the width of the character bounding box.

3) Character Barycentre Coordinate Feature (Bx, By): Starting from the motivation that different character shapes have different barycentre positions, as shown in Figure 4, we propose a novel and simple barycentre computation technique. Indeed, we consider the polygon composed only with the starting segment pixels. Then, we compute the x and y coordinates of the corresponding barycentre for both Horizontal and vertical Merged CSD Vectors.

4) Character Ratio Feature (R): To improve the discrimination between characters that have similar number of segments and Barycentre position, we propose to compute

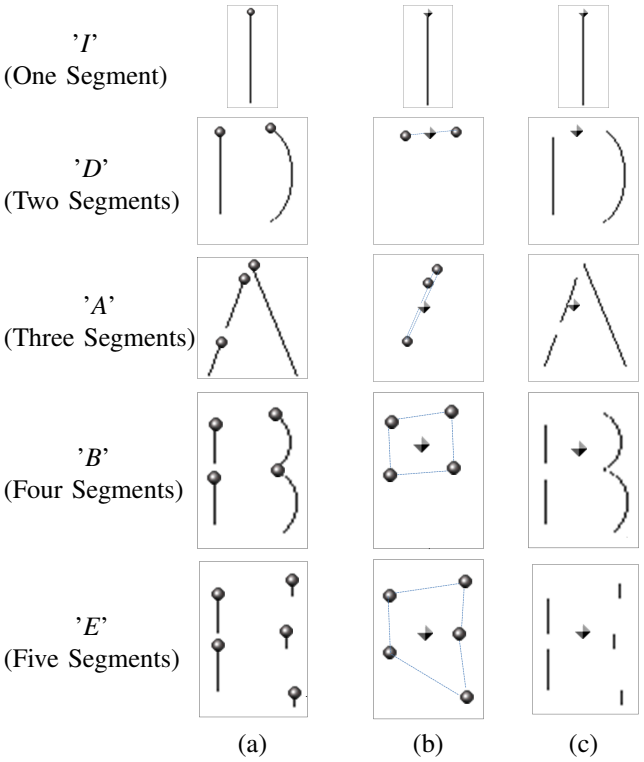


Fig. 4: Barycentre (B) computation process on different segment number letters: a. Starting pixels, b. Barycentre of polygon computation, c. Barycentre result

the Ratio between the height and width of the processed character bounding box. This feature is computed according to the Equation (5).

$$R = \frac{w}{h} \quad (5)$$

Figure 5 shows a Multiscale representation of characters '*A*' and '*J*'. As we can see, the Ratio feature (R) does not changes for the same character with different sizes (Equation (6)):

$$\begin{cases} R('A') = \frac{w_{00}}{h_{00}} = \frac{w_{01}}{h_{01}} = \frac{w_{02}}{h_{02}} = \frac{w_{03}}{h_{03}} = \frac{w_{04}}{h_{04}} = \frac{w_{05}}{h_{05}} \\ R('J') = \frac{w_{10}}{h_{10}} = \frac{w_{11}}{h_{11}} = \frac{w_{12}}{h_{12}} = \frac{w_{13}}{h_{13}} = \frac{w_{14}}{h_{14}} = \frac{w_{15}}{h_{15}} \end{cases} \quad (6)$$

However, in the same scale, the Ratio feature (R) allows to discriminate easily the two different characters '*A*' and '*J*' (Equation (7)):

$$\begin{cases} Scale_0 : [R('A') = \frac{w_{00}}{h_{00}}] \neq [R('J') = \frac{w_{10}}{h_{10}}] \\ Scale_1 : [R('A') = \frac{w_{01}}{h_{01}}] \neq [R('J') = \frac{w_{11}}{h_{11}}] \\ \vdots \\ Scale_5 : [R('A') = \frac{w_{05}}{h_{05}}] \neq [R('J') = \frac{w_{15}}{h_{15}}] \end{cases} \quad (7)$$

In fact, the Ratio feature (R) ensures a Multiscale invariance and increases the discrimination between characters in the

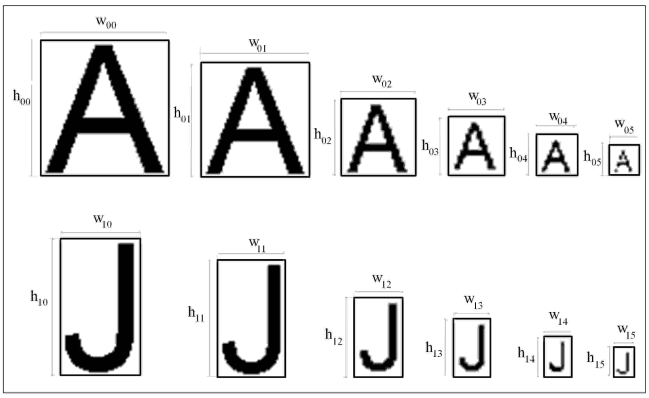


Fig. 5: Scale invariance of the character ratio feature of '*A*' and '*J*' letters with six different scales

same scale. Based on these extracted features from Horizontal and Vertical segments, the Unified Character Descriptor (UCD) is computed. Indeed, Horizontal and Vertical features are organized in a subsequent way in one single UCD vector as illustrated in Figure 6. Except the Barycentre x and y coordinates are computed according to the average of Horizontal and Vertical ones from the CSD vectors.

Number of Segments		Segment Position: Top, Middle, Bottom		Segment Position: Left, Middle, Right		Barycentre		Ratio
H-NBS	V-NBS	H-TMB	V-TMB	H-LMR	V-LMR	$\overline{Bx}$	$\overline{By}$	H-R
(a)		(b)				(c)		(d)

Fig. 6: The Unified Character Descriptor (UCD): a. Character Segment Number Feature (NBS), b. Character Segment Position Feature (TMB, LMR), c. Character Barycentre Feature ($\overline{Bx}$, $\overline{By}$), d. Character Ratio Feature (R)

B. The Template Selection Stage

In our proposed method, the template selection stage is performed through type-face and font-size recognition. Indeed, even if the above UCD is scale invariant, it stills more powerful on small size ranges. To improve the Multiscale characteristic of our method, we propose to compute effectively the character font-size to choose the appropriate template to use. In addition, for an identified type-face character it is possible to make the OCR system handle a document with less effort. Following we detail our template selection proposal.

1) Font-Size Recognition (x-h): Text line documents are composed of three typographical zones: the ascender, the x-height and the descender zones [16]. As shown in Figure 7, these zones are delimited by four virtual horizontal lines, Ascender, x-height, Base and Descender lines. The x-height

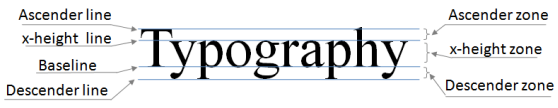
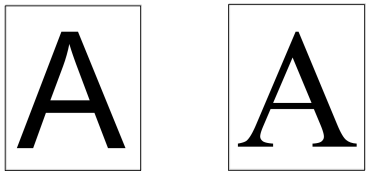


Fig. 7: Typography of Latin Alphabets

zone is the height of the body of lowercase letters referring to the distance between the baseline and the x-height one of lower-case letters in a typeface. For Latin script the Font-Size recognition is relatively an easy task [18][14]. In our proposal, we identify the character size with the help of the computation of the connected components x-h. Hence, we apply the following formula shown in Equation (8) [18].

$$\text{x-h} = \text{Baseline} - \text{x-height line} \tag{8}$$

2) Type-Face Recognition (σ): Type-face recognition can give details on the structural and the typographical design of characters. Type-faces can be divided into two main categories: *serif* and *sans serif*. We mean by serifs the small features at the end of strokes within letters. Printed type-faces without serifs are known as *sans serif*. '*A*' letter with *serif* and *sans serif* type-faces is shown in Figure 8.



(a) (b)

Fig. 8: '*A*' letter with: a. *Sans Serif* and b. *Serif* type-faces

Actually, it is assumed that serif type-faces contain characters with moderate or dramatic difference between thick and thin strokes [17]. However sans serif characters are characterized with a low difference between stroke thicknesses. For this, we propose to study the Standard Deviation (σ) across the different character strokes (Equation (9)).

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu)^2}; \quad \mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{9}$$

where x is the measured thickness in one character block, N is the number of character strokes and μ is the average of the N measured thicknesses; $i \in [1, N]$. To do this, we divide the character image generated by the layout analysis stage into 3x3 blocs.

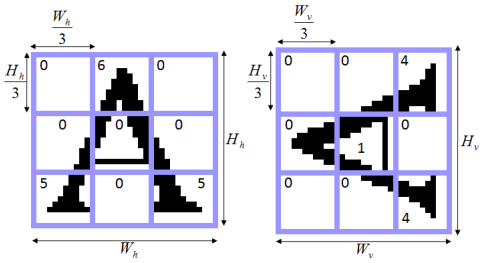


Fig. 9: Font recognition based on the splitting of characters in equal size blocks. Each block number refers to the median thickness of the considered non overlapping strokes

As shown in Figure 9, we determine in each block the thickness as the median of non overlapping lines with the block edges. During this process, vertical (stem, hairline, etc) and horizontal strokes (cross stroke, cross bar, etc) are checked. Very long or short lines (such as brackets, apex) that do not give a significant indication regarding the thickness are eliminated thanks to the 3x3 bloc division. As illustrated in Figure 10, the measured stroke thickness Standard Deviation (σ) is approximately constant between different Font-Sizes however it varies considerably between different type-faces.

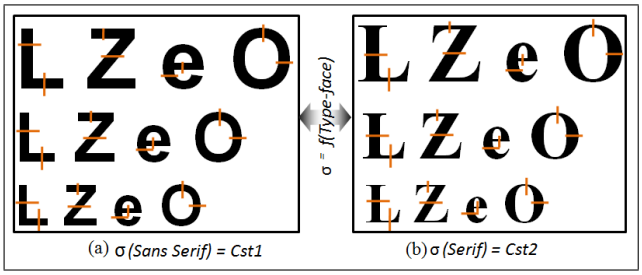


Fig. 10: Standard Deviation (σ) of stroke thickness in Multi-scale and Multifont characters. a. *Sans Serif* with no/moderate transition in the strokes, b. *Serif* with dramatic thick/thin transition in the strokes

C. The Decision Stage

The final stage of our work consists on matching the UCD vector of the input character to the different UCD vectors of the selected template (see Figure 2). The matching is performed by using the Sum of Absolute Differences (SAD) technique [12] distance process.

III. Experimental results

In this section, we evaluate firstly our proposed UCD descriptor regarding the robustness against the 'i' algorithm [9] descriptor when dealing with Multifont and Multisize constraints. Then, we evaluate our CR-UCD system against the well known Tesseract 3.02 OCR engine [20] on the same previously mentioned constraints.

A. The UCD Descriptor Evaluation based on OCR Accuracy

To evaluate the OCR-accuracy of our proposed UCD descriptor we use Multiscale and Multifont computer generated alphanumeric images containing number, upper-case, lower-case and special characters. For comparison reasons, we assess also the recently proposed 'i' Algorithm [9]. Table 1 shows a comparison between UCD and 'i' Algorithm descriptors using OCR Accuracy. In this evaluation, each character is matched with the same character font and size in the template. Obtained results demonstrate that our method outperforms the 'i' algorithm reaching an average of 99% of OCR accuracy on all assessed font-sizes and Type-faces. The main issues encountered by our description is the miss-recognition of

Table 1: OCR Accuracy of UCD and 'i' Algorithm descriptors using Multifont and Multisize template

Method	Type-face	Font-Size				
		18	24	36	72	90
'i' Algorithm	Serif	90	78	90	95	98
	Sans Serif	80	77	82	93	95
CR-UCD	*Serif*	**100**	**100**	**100**	**100**	**100**
	Sans Serif	**98**	**98**	**98**	**98**	**98**

similar characters like 'I' and 'l' in the different scales of the *sans serif* Type-face. In future work, lexical methods can be used in that case to enhance the performance of our description.

In Table 2 we illustrate the performance of UCD and the 'i' Algorithm descriptors using one single template defined by the *serif* type-face and the 90pt font-size for all compared character fonts and sizes. We can see that in this case the

Table 2: OCR Accuracy of UCD and 'i' Algorithm descriptors using one single font and size template

Method	Type-face	Font-Size				
		18	24	36	72	90
'i' Algorithm	*Serif*	12	20	25	59	98
	Sans Serif	15	14	20	50	95
CR-UCD	Serif	31	50	52	91	**100**
	Sans Serif	25	30	44	70	**98**

UCD still outperforming the 'i' algorithm descriptor in *serif* and *sans serif* fonts. Moreover, we note that the recognition of the *serif* text is better than the *sans serif* one. Indeed, characters 'I' and '1' are distinguished in the *serif* text thanks to their different morphology. However, the similarity of these characters in the *sans serif* text prevents their right recognition. Despite that our proposed descriptor gives a high accuracy on single type-face and font-size, the OCR Accuracy drops down when dealing with Multifont and Multiscale data. Hence, the usefulness of the template selection process in the CR-UCD System.

B. The CR-UCD System Evaluation on the LRDE Dataset

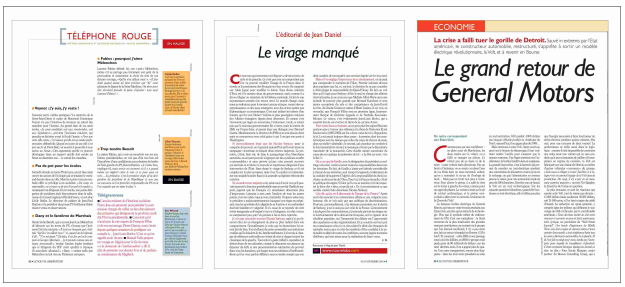


Fig. 11: Sample of the LRDE-DBD documents

Following, we evaluate our CR-UCD proposal when pro-

cessing on the LRDE-DBD dataset[1] [19] composed of 125 magazine documents. As shown in Figure 11, the LRDE-DBD includes different font-sizes, *serif* and *sans serif* type-faces. We show in Figure 12 the employed templates used in the Feature matching. We generate six templates categorized into two classes within *serif* and *sans serif* type-faces. Each category, includes three sub-templates consisting of three scale ranges: small, medium, and large character sizes.

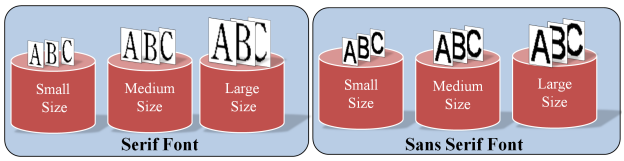


Fig. 12: The employed Multisize and Multifont templates

Small characters refer to the core paragraph text, medium characters refer to subtitles and large size characters are considered as the titles. Empirically fixed, the employed thresholds to distinguish font-sizes are given in Table 3.

Table 3: Employed x-height thresholds

Character sizes	x-height thresholds (pixels)
Small	0-30
Medium	30-55
Large	55-100

To show a relevant evaluation, we compare our proposed CR-UCD OCR system to the Tesseract [20] one. To make a fair comparison, we disabled the Tesseract dictionary option that we did not handle yet in our work. Figure 13 shows that the proposed CR-UCD method outperforms Tesseract one by reaching 1.5x higher OCR Accuracy on the 125 magazine documents of the LRDE Dataset.

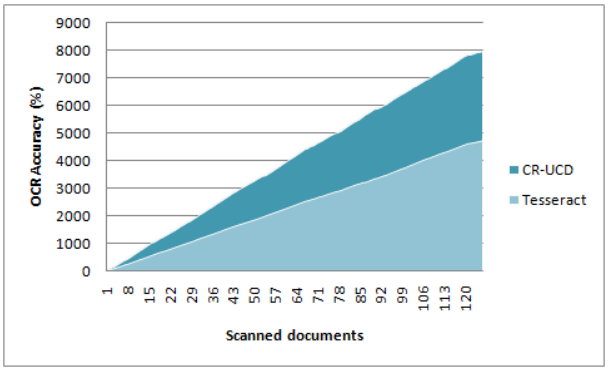


Fig. 13: OCR Accuracy evaluation of Tesseract and the proposed CR-UCD method

[1]Copyright(c) 2012. EPITA and Development Laboratory (LRDE) with permission from Le Nouvel Observateur. LRDE-DBD is available on-line on the web site: $http : //www.lrde.epita.fr/cgi - bin/twiki/view/Olena/DatasetDBD$

IV. Conclusion and Future Work

In this paper, we proposed an efficient OCR system based on a new and robust description. Thanks to the used UCD descriptor, the template selection, and the employed fast feature matching our proposal offers a high OCR Accuracy compared to well known description and recognition methods. In future work, we intend to improve the proposed OCR efficiency by using additional semantic rules to be able to differentiate similar characters like 'I' and 'l'. In addition, the template selection process could be enhanced to distinguish thick and italic characters. We plan also to use dictionary to enhance the final recognition results.

References

[1] J.R. Prasad, Template matching algorithm for Gujrati Character Recognition, IEEE 2nd International conference on Emerging Trends in Engineering and Technology, ICETET (2009)

[2] Z.You-qing, A Recognition Method of Car License Plate Characters Based on Template Matching Using Modified Hausdorff Distance. IEEE International Conference on Computer, Mechatronics, control and electronic Engineering (CMDE) (2010)

[3] F.Aghdasi, Automatic Licence Plate recognition system, IEEE AFRICON (2004)

[4] J.Jiao, A configurable method for multi-style license plate recognition. Pattern Recognitionn, Vol: 42, No: 3, pages 358-369 (2009)

[5] M.M Farhad, An efficient Optical Character Recognition Algorithm using Aritificial Neural Network by Curvative Properties.IEEE International conference on informatics, electronics é vision (2014)

[6] M.soua, R.Kachouri, A.Akil. A new hybrid binarization method based on Kmeans, ISSCP 2014.

[7] S.Kopf, T.Haenselmann, W.Effelberg. Robust Character Recognition in Low-Resolution Images and videos.

[8] A. Antanacopoulos, C. Clausner, C. Papadopoulos, and S.Pletschacher. ICDAR2013 Competition on Historical Newspaper Layout Analysis., International Conference on Document Analysis and Recognition, 2013

[9] S.Sharstry, Gumasheela, T.DUTT, D.S.Vinay. "i" A novel algorithm for optical character recognition (OCR), 2013.

[10] R. Smith, *"Hybrid Page Layout Analysis via Tab-Stop Detection"*, ICDAR '09 Proceedings of the 2009 10th International Conference on Document Analysis and Recognition, Pages 241-245, 2009

[11] R. Smith, *"An Overview of the Tesseract OCR Engine"*, Proc. Ninth Int. Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society, 629-633 , 2007.

[12] K.Lengwehasarit, *"Probabilistic partial-distance fast matching algorithms for motion estimation "*, pp 139 - 152. Circuits and Systems for Video Technology, IEEE Transactions on (Volume:11 , Issue: 2), 2002.

[13] M.Soua, R.Kachouri, M.Akil. GPU parallel implementation of the new hybrid binarization based on Kmeans method (HBK) Journal of Real-Time Image Processing, Springer, 2014

[14] Q.Akram, S.Hussain, Z.Habibt, Font-Size Independent OCR for Noori Nastaleeq.

[15] http://www.infor.uva.es/ descuder/docencia/IG/letterform_anatomy.pdf

[16] Z. Abdelwahab and R. Ingold, Optical Font Recognition Using Typographical Features,IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 8, pp.877-882, 1998

[17] http://www.sitepoint.com/the-old-style-type-face/

[18] Souza, M. ; Naoi, S. ; Suen, C.Y. A.Automatic filter selection using image quality assessment, 2003

[19] G. Lazzara, T. Géraud, *"Efficient Multiscale Sauvolas Binarization"*, International Journal on document analysis and recognition, 2013

[20] R. Smith, An Overview of the Tesseract OCR Engine. International Conference on Document Analysis and Recognition, 2007.