

Reading Skewed Images without Image Rotation

Abstract— In this paper, we will present a new approach for extracting characters from a skewed images without image rotation. Our approach is capable of handling any font and size of characters. The proposed method consists of sorting the segmented characters according to the X axis, then assigning a Program Line Number to each character based on the skew angle and finally sorting the Program Line Numbers according to their intersection with the Y axis. This new method of characters extraction can deal from -80° to 80° degree of image rotation with an average processing time around 0.11 milliseconds.

Index Terms— Image Rotation, Skew angle, Container Number.

1. INTRODUCTION

Due to the incurred problems related to rotated images taken from any real-life scenario, many papers tackled this issue in order to detect and correct the skewed images. The existing algorithms consist of rotating the image after calculating its angle inclination [1-3] before proceeding into the next step of the Optical Character Recognition (OCR) process.

On the one hand, image rotation can never add details to the image and thus the result of an adjustment of a skewed image can easily deteriorate the quality of the image (colors less intense, haloes creation around the characters) as illustrated in Fig.1 and Fig.2 and may also result in losing some of the surrounding pixels and/or characters as shown in Fig.3 and Fig.4.



Figure 1. Container number original image.

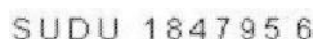


Figure 2. Quality deterioration plus haloes after two 25° rotations.

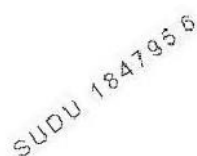


Figure 3. Container number skewed image



Figure 4. Lost of pixels/characters after 25° rotation.

On the other hand, image rotation is time consuming due to the complexity of its computation which is considered as a big problem in real-time systems because it affects the overall character recognition process and therefore it is indispensable to avoid it.

2. PROPOSED METHOD

Based on the above described drawbacks of image rotation, our proposed method consists of extracting the characters of an image subject to a later Optical Character Recognition (OCR) without rotating the image. Our approach is very simple and consists of the following two phases: preparatory and execution phases.

A. Preparatory Phase

The preparatory phase consists of relying on any existing characters' segmentation and determination of the image rotation angle.

A.1 Characters' segmentation

Any segmentation approach can be used in order to detect the useful segments from a rotated image [4-11], and all we need is the upper-left coordinates represented by $(X_{\min}, Y_{\min})$ and the lower-right coordinates represented by $(X_{\max}, Y_{\max})$ of each segment that will be used in the OCR process. In this paper, a character will be represented in the following way $(X_{\min}, Y_{\min}, X_{\max}, Y_{\max})$.

It is worthy to note that in digital image processing, images are usually read from top to bottom and from left to right, so as shown in Fig. 5, the X and Y axes are a little bit different than the normal used ones.

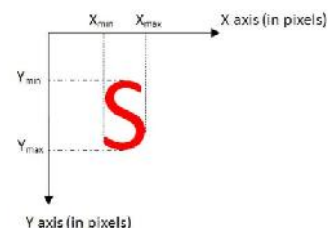


Figure 5. Letter S coordinates.

A.2 Image rotation angle calculation

Any skew detection approach can be used in order to determine the rotation angle of the image (represented hereafter by θ), worthwhile to mention that such algorithms are classified into the following four main categories [12]: Hough transform [13,14], projection profile [15,16], nearest neighbor clustering [17,18] and interline cross correlation [19-21].

All we need from this sub-phase is to know the image rotation angle in order to start the execution of our proposed approach.

B. Execution Phase

The execution phase is the kernel of our approach and is divided into Segments Line Determination and Program Lines Ordering.

B.1 Segments Line Determination

In order to acquire the best response time, our approach consists working all the time in the memory instead of going to any other computer peripheral.

First, we start by determining the coordinates (X, Y) of the center of each segmented character retrieved in the preparatory phase, in the following way:

$$X = (X_{\min} + X_{\max}) / 2$$

$$Y = (Y_{\min} + Y_{\max}) / 2$$

where $(X_{\min}, Y_{\min})$ represents the upper-left corner of the segmented character and $(X_{\max}, Y_{\max})$ represents the lower-right corner of the segmented character.

Second and because Latin characters are written from left to right, we will sort the segmented characters according to their $X_{\min}$ value and thus there will be no problem if the segmented characters which were retrieved in the preparatory phase are scrambled (i.e. retrieved in any order).

Third, we initialize the Program Line Number (PLN) variable by zero then we enter the first loop where the variable "I" will hold the first till the last segment of the previously sorted array, where we will:

- a. Check if the segment I having –for example- the center (X_1, Y_1) , was previously assigned to a Program Line Number; if the answer is positive we will pass to the next step of the first loop (i.e. we put in I the next segment of the sorted array and we go back to step a), otherwise we will increment the PLN by one and we assign it to I and then we will enter a second loop where the variable "J" will holds the segment after I till the last segment of the sorted array, where we will:
- b. Check whether the segment J was previously assigned with a Program Line Number; if the answer is positive we will pass to the next step of the second loop (i.e. we put in J the next segment of the sorted array and we go back to step b), otherwise we check whether the Imaginary Line (represented hereafter by IL) passing by (X_1, Y_1) crosses the segment J; if the answer is positive, PLN is assigned to J otherwise nothing will be done. Finally, we will pass to the next step of the second loop (i.e. we put in J the next segment of the sorted array then we go back to step b).

It is important to note that:

1. If an image is rotated by θ , all the lines inside the image will be rotated by the same value.
2. The slope of the line passing by the two points (X_1, Y_1) and (X_2, Y_2) is equal to $(Y_2 - Y_1) / (X_2 - X_1)$ which is equal to the tangent of θ .

3.

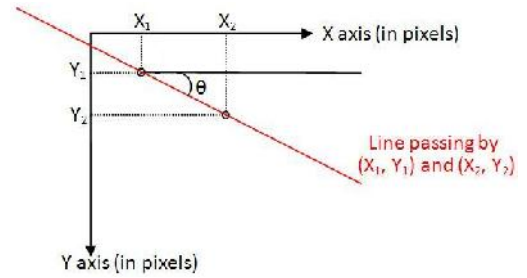


Figure 6. Relation between angle rotation and line slope

4. The equation of the imaginary line passing by (X_1, Y_1) and having the slope "S" is equal to:
 $Y = S(X - X_1) + Y_1$;
5. The Imaginary Line passing by (X_1, Y_1) and having the slope S crosses a segment if it passes by/below its upper-left coordinates $(X_{\min}, Y_{\min})$ and by/above its lower-right coordinates $(X_{\max}, Y_{\max})$, which means:
 $Y_{\min} \leq S(X_{\min} - X_1) + Y_1$ and
 $Y_{\max} \geq S(X_{\max} - X_1) + Y_1$

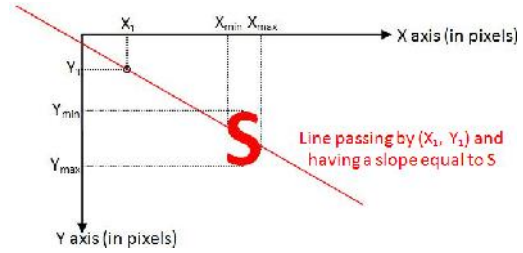


Figure 7. Example of a line crossing a segment

6. All the segments which are crossed by the Imaginary Line passing by (X_1, Y_1) and having the slope S will have the same PLN because they are in the same line of the rotated image.
7. At the end of the B1 phase, all the segments will be assigned to a PLN but the PLNs will not reflect the real ordering of the image lines.

B.2 Program lines ordering

At this stage, all the segments are classified according to their PLNs but which may not be similar to the real order of the image lines because our approach can deal with scrambled segments. To solve this issue, the PLNs will be sorted according to their Imaginary Line intersection with the Y axis. Below, the three figures (Fig. 8, 9 and 10) will illustrate the problem and its solution.

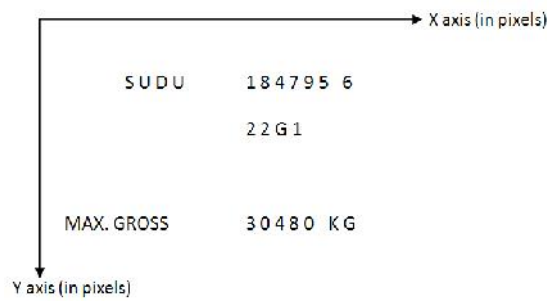


Figure 8. Original binarized upper-right back container image.

Fig.9 shows the PLNs state before the Program Line Ordering phase (i.e. this problem occurred due to sorting the segments according to their $X_{\min}$ value)

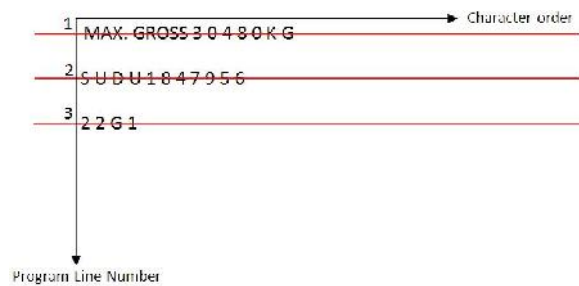


Figure 9. PLNs before Program Lines Ordering phase

Fig.10 shows the PLNs ordered according to their Imaginary Line intersection with the Y axis.

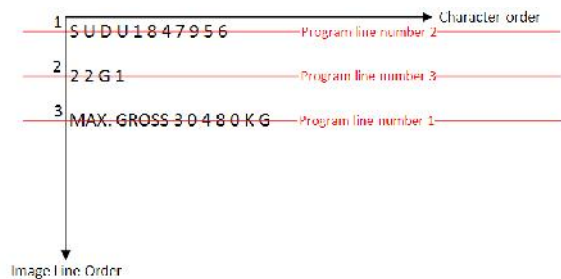


Figure 10. PLNs after the Image Line Ordering phase

3. PROPOSED METHOD FLOWCHART

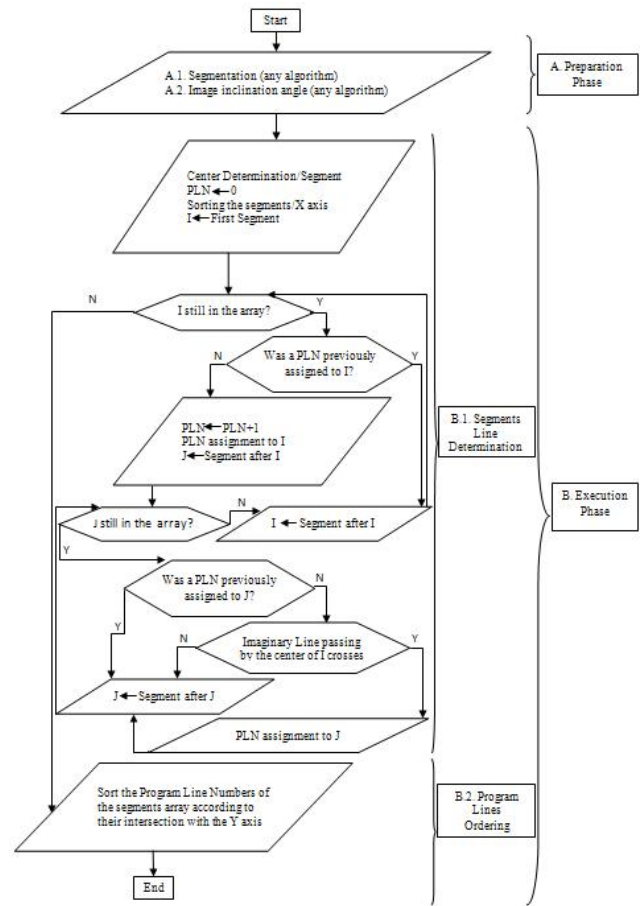


Figure 11. Flowchart of the whole proposed method

4. EXPERIMENTS

Our proposed method was initially designed for skewed images taken from the container back side and it succeeded to return very good results regarding its performance and ability to deal with high skewed angles.

The result of our experiments showed a response time around 0.11 milliseconds for executing our approach on an image containing the container number and the container type while using a laptop having a 2.00 GHz processor and 2 GB of RAM, which is highly desirable for real-time system.

It is essential to note that the existence of spaces before or in the middle of the container number or container type doesn't have any negative effect on the result.

Below, we will illustrate some samples from our experimental results. To make the figures more readable, each figure will contain two parts where the part above the red line represents the original image which contains two lines (the first one represents the Container Number and the second one represents the Container Type) and the part below the red line represents the extracted segments without applying any image rotation.

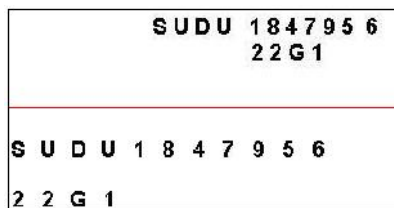


Figure 12. Image rotation = 0°

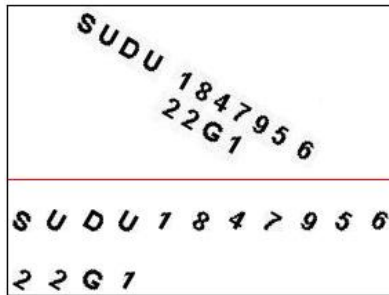


Figure 13. Image rotation = -30°

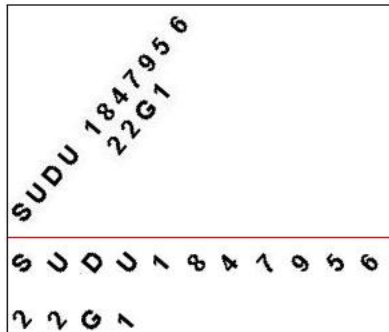


Figure 14. Image rotation = 50°

Worthwhile mentioning that the inclination of the extracted segments doesn't represent any issue because all the OCR applications can easily read them after being trained.

Finally, we would like to highlight that our proposed method can tolerate and deal with distorted segments, due to image bad quality or subject to image filtering, because we designed our approach in a way that a segment is affected to a PLN if the Imaginary Line crosses it (i.e. disregarding if it is a little bit higher or lower from the center of the segment).

5. APPLICATION FIELD

Although this paper was intended to read the container number of a skewed image (in the front –Fig. 15-, back –Fig. 16-, left, right or top side of the container), it can also be used to read the container owner name –Fig. 17- and the vehicle plate numbers –Fig. 18-.



Figure 15. Container front side



Figure 16. Container back side



Figure 17. Image showing the owner of the container



Figure 18. Vehicle plate

6. COMPARATIVE STUDY

In this study, we will present the benefit from being able to extract segments from a skewed image without rotating it; to

this end, below we will list the execution time for image deskewing as mentioned by their authors:

- The best computation time for skew correction is 0.017 second for a resolution equal to 0.45 Mega pixel for a 800x600 image [22].
- The time cost varies from 200 to 1000 millisecond [23].
- Around 14.7 millisecond is needed to rotate 256x256 image [24].
- The average processing time is 77.998 millisecond if implemented on a CAM-based architecture and 128.367 millisecond when implemented on RAM-based Architecture [25].
- The average processing time is between 17-110 milliseconds for 0.45-3.0 mega pixel images [26].

7. FUTURE WORKS

Because our approach was intended for extracting the container number mainly from the back side of the container, so it was designed to read any font and letter size -but having the same height- and thus further enhancements should be done to deal with capital and small letters belonging to the same line; the second issue that needs further research is to calculate, from the segmented characters, the image rotation angle without relying on any other existing algorithm because they are not 100% accurate while detecting the rotation angle of an image and thus the result may sometimes be incorrect.

Fig.19 illustrates an example of an inaccurate angle rotation value which led to an incorrect result because the container number was split into two lines. The red lines in the Fig.19 represent the Imaginary Lines where the upper part of the image represents the original image and the lower part represents the result.

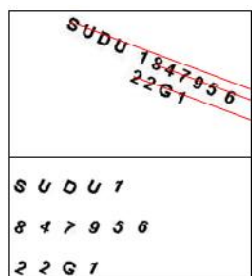


Figure 19. Incorrect angle rotation

Finally, we will list the extract of the error deviation as mentioned by their authors:

- The average deviation error is estimated within two degrees from the true angle [27]
- The Rotation from -5° to 5° presumed an error deviation of 0.2° [12]
- The average deviation error of the skew corrected text lines is from -3° to 3° [22]
- The skew angle error varies from 0° to 0.6° [23]

- The skew angle error is between -3° and 3° degrees [26]

REFERENCES

- [1] T.D. Duan, T.L.H. Du, T.V. Phuoc and N.V. Hoang, "Building an Automatic Vehicle License-Plate Recognition System", International Conference in Computer Science, RIVF'05, 2005.
- [2] J.C.M. Lee, "Automatic character recognition for moving and stationary vehicles and containers in real-life images", International Joint Conference on Neural Networks, IJCNN '99, Vol. 4, 1999.
- [3] A.F. Mollah, N. Majumder, S. Basu and M. Nasipuri "Design of an Optical Character Recognition System for Camera-based Handheld Devices", IJCSI International Journal of Computer Science Issues, Vol. 8, No 1, 2011.
- [4] H. Mobahi, S. Rao, A. Yang, S. Sastry and Y. Ma "Segmentation of Natural Images by Texture and Boundary Compression", International Journal of Computer Vision (IJCV), 95 (1), pg. 86-98, 2011.
- [5] K J. Batenburg and J. Sijbers, "Adaptive thresholding of tomograms by projection distance minimization", Pattern Recognition, vol. 42, no. 10, pp. 2297-2305, 2009.
- [6] P. Arbelaez, M. Maire, C. Fowlkes and J. Malik "Contour Detection and Hierarchical Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 33, 2011.
- [7] Z. Ren, G. Shakhnarovich "Image Segmentation by Cascaded Region Agglomeration", CVPR, 2013.
- [8] R. Achanta et Al. "SLIC superpixels compared to state-of-the-art superpixel methods", IEEE TPAMI, 2012.
- [9] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik "Contour detection and hierarchical image segmentation", IEEE TPAMI, 33(5), 2011.
- [10] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu "Semantic segmentation with second-order pooling", In ECCV, 2012.
- [11] V. Karthikeyan, R. Sindhu, K. Anusha and D.S. Vijith, "Vehicle License Plate Character Segmentation – A STUDY", International Journal of Computer and Electronics Research, Vol. 2, 2013.
- [12] A. Papandreou and B. Gatos, "A Novel Skew Detection Technique Based on Vertical Projections", International Conference on Document Analysis and Recognition, 2011.
- [13] X. Fan, G. Fan and D. Liang "Joint Segmentation and Recognition of License Plate Characters", IEEE International Conference on Image Processing, ICIP, 2007.
- [14] C. Singh, N. Bhatia and A. Kaur, "Hough transform based fast skew detection and accurate skew correction methods", Pattern Recognition, vol.41, pp. 3528-3546, 2008.
- [15] J. Jagannathan, A. Sherajdeen, R.M.V. Deepak and N. Krishnan "License plate Character Segmentation using horizontal and vertical projection with dynamic thresholding", International Conference on Emerging Trends in Computing, Communication and anotechnology (ICE-CCN), 2013.
- [16] N. Liolios, N. Fakotakis and G. Kokkinakis "Improved document skew detection based on text line connected component clustering". International Conference on Image Processing, volume 1, pp. 1098-1101. Proceedings. 2001.

- [17] Sarfraz M., Mahmoud S.A. and Rasheed Z. "On Skew Estimation and Correction of Text", Computer Graphics, Imaging and Visualisation, CGIV '07, 2007.
- [18] Y. Lu and C.L. Tan "A nearest-neighbor chain based approach to skew estimation in document images". Pattern Recognition Letters, Vol.24, pp. 2315–2323, 2003.
- [19] T. Akiyama and N. Hagita; "Automatic entry system for printed documents," Publisher Elsevier Science, Pattern Recognition vol. 23 , issue 11, pp.1141 – 1154, 1990.
- [20] M. Chen and X. Ding "A robust skew detection algorithm for grayscale document image", Proceedings of the Fifth International Conference on Document Analysis and Recognition, ICDAR '99, 1999.
- [21] A. Chaudhuri, S. Chaudhuri "Robust detection of skew in document images", IEEE Transactions on Image Processing, vol.6, pp.344–349, 1997.
- [22] A.F. Mollah, S. Basu, N. Das and R. Sarkar, "A Fast Skew Correction Technique for Camera Captured Business Card Images ", IEEE Annual India Conference (INDICON), 2009.
- [23] Y. Zhongda, D. Junyu, W. Zhiqiang and S. Jianxiang "A Fast Image Rotation Algorithm for Optical Character Recognition of Chinese Documents ", International Conference on Communications, Circuits and Systems Proceedings, 2006
- [24] Xiao-Gang JIANG, Jian-Yang ZHOU, Jiang-Hong SHI and Hui-Huang CHEN "FPGA Implementation of Image Rotation Using Modified Compensated CORDIC", 6th International Conference on ASIC Proceedings, ASICON, 2005.
- [25] Y.-P. Chen, T.-D. Yeh, F.-C. Ni and C.-H. Chang, "A CAM-Based Pattern Accumulated Vector Method for Real-Time Character Recognition of License Plates", IJCSNS International Journal of Computer Science and Network Security, Vol. 9 No. 1, 2009.
- [26] A.F. Mollah, S. Basu, N. Das and R. Sarkar, "A Fast Skew Correction Technique for Camera Captured Business Card Images ", Annual IEEE India Conference, INDICON, 2009.
- [27] H. Hase, K. Tanabe, T.H.H. Tran and S. Tokai "Multi-Font Rotated Character Recognition Using Periodicity", The Eighth IAPR International Workshop on Document Analysis Systems, DAS'08, 2008.