2018 24th International Conference on Pattern Recognition (ICPR)
Beijing, China, August 20-24, 2018

Scene text rectification using glyph (字形) and character alignment properties

Taeho Kil
Dept. of Electrical and Computer Eng.
Seoul National University
Seoul, Korea
Email: xellows1305@ispl.snu.ac.kr

Hyung Il Koo
Dept. of Electrical and Computer Eng.
Ajou University
Suwon, Korea
Email: hikoo@ajou.ac.kr

Nam Ik Cho
Dept. of Electrical and Computer Eng.
Seoul National University
Seoul, Korea
Email: nicho@snu.ac.kr

***Abstract*—Scene text images usually suffer from perspective distortions, and hence their rectification has been an essential pre-processing step for many applications. Existing methods for scene text rectification mainly exploited the glyph property, which means that the characters in many languages have horizontal/vertical strokes and also have some symmetries in their shapes. In this paper, we propose to use an additional property that the characters need to be well aligned when rectified. For this, character alignment, as well as glyph properties, are encoded in the proposed cost function, and its minimization generates the transformation parameters. For encoding the alignment constraints, we perform the character segmentation using a projection profile method before optimizing the cost function. Since better segmentation needs better rectification and vice versa, the overall algorithm is designed to perform character segmentation and rectification iteratively. We evaluate our method on real and synthetic scene text images, and the experimental results show that our method achieves higher optical character recognition (OCR) rate than the previous approaches and also yields visually pleasing results.**

I. Introduction

Understanding the text in an image is important for many applications. Hence, numerous methods have been proposed for the detection and recognition of the text in the wild scene (scene text) [1]–[4]. However, the Optical Character Recognition (OCR) of camera-captured images is still considered a challenging problem, even after the successful text detection. It is mainly due to the perspective distortion caused by camera pose, and most OCR systems at present cannot manage this distortion. Hence, the rectification is an essential pre-processing step for the recognition of scene text.

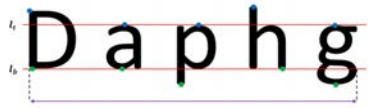
There have been many methods for the text rectification in the case of document images [5]–[12], where the images contain lots of text and text-lines. Many scene text rectification methods have also been separately developed [13]–[22], the approaches of which are quite different from those for the document images. In general, the difficulties in scene text rectification are summarized as follows. First, the scene text image contains a few characters compared to the document image. We can extract much valuable information in the case of text-abundant document images, whereas there are few features to be extracted in the case of scene text. Second, there are too many variants in the character shapes, (mixed) languages and stroke widths. Hence, it is difficult to rectify the scene without the prior information. Finally, dealing with the perspective distortion is not straightforward. Generally, when the text is on a planar surface, its transformation to the fronto-parallel view can be modeled as a projective transformation. In this transformation, there are four parameters related to skew, shearing, horizontal and vertical foreshortening, where estimating these parameters simultaneously is a challenging problem.

(a) (b) (c)

Fig. 1: Limitations of the cost function using only glyph (low-rank) property. (a) Input images. (b) Images minimizing the cost function in [16]. (c) Well rectified images. For example, the low-rank costs of images on the bottom row are 46.3, 46.1, 47.9, respectively.

Considering the problems related to these difficulties, we propose a new algorithm that exploits alignment constraints in addition to other properties employed in the conventional works. Specifically, the existing methods for scene text rectification mainly exploited the glyph property which is a common shape property for the undistorted characters. Some examples of glyph properties are that the characters in many languages have horizontal and/or vertical strokes and many characters have some symmetries in their shapes. When the character or set of characters is represented as a matrix, the rank of the matrix for the well-rectified character is usually lower than that for the distorted ones.

However, the glyph-based methods sometimes yield misaligned results for multiple characters. For example, as shown in Fig. 1, the cost function in low-rank transform [16] has minimal value even though the image is not well rectified because its cost function yields small values for not only structured (low-rank) regions but also for a blank space. Also, the low rank does not necessarily mean the alignment. For alleviating



(a)

(b)

Fig. 2: Illustration of the character alignment properties. (a) Many of top (blue) and bottom (green) points of characters are on two (red) alignment lines. (b) The width of a well-rectified character has the minimal value. We encode these properties into the cost function.

this problem, we propose to use character alignment properties in addition to the conventional glyph property. Specifically, we design a cost function including these two properties, the minimization of which provides the transformation parameters. To encode the alignment properties into the cost function, we need to segment the text into individual characters. The character segmentation is relatively easy when the text is in fronto-parallel view without any perspective distortion, but it is also a difficult problem in the case of distorted images [23], [24]. In short, the rectification and character segmentation are a chicken-and-egg problem, i.e., better segmentation needs better rectification and vice versa. We solve this problem by performing the character segmentation and rectification iteratively.

In summary, we develop a cost function consisting of the conventional glyph term (low-rank constraint) and the proposed alignment term. We believe that the main contributions of this paper are the development of a new cost function that deals with both of glyph and character alignment properties for rectification, and the method of iterating character segmentation and rectification.

II. Proposed method for rectification

Similar to the previous methods [13]–[20], we assume that the scene text is on a planar surface and the text regions are already detected and binarized so that we focus on the rectification only. Then, the rectification process is formulated as a minimization problem that estimates the parameters of projective transformation. The cost function consists of glyph term and character alignment term, where we adopt the low-rank measure for the glyph term according to [16]. Hence we focus on the explanation of alignment term in the rest of this section.

A. Cost function design

Most of the scene text, except for artistic rendering of text, are vertically or horizontally aligned. As shown in Fig. 2, vertical alignment means that characters in the undistorted text are aligned to the horizontal lines, specifically many of top and bottom points of characters are on one of two lines (top line l_t and bottom line l_b as denoted in Fig. 2(a)). The horizontal alignment means that the characters in the undistorted text have minimal width as the center 'A' in Fig. 2(b).

In order to encode these alignment properties into the cost function, we first introduce a set of alignment parameters $\mathcal{W} = \{(\omega_i, \mu_i)\}_{i=1}^{N}$, where ω_i and mu_i are binary variables. Specifically, ω_i represents whether the top point of the i-th character is aligned to the top line l_t, and μ_i denotes whether the bottom point is aligned to the bottom line l_b. For instance, the top point of 'D' in Fig. 2 is not on the l_t and the bottom point is on l_b, then (ω_i, μ_i) for this character is $(0, 1)$.

For the estimation of transformation parameters τ, we develop a cost function:

$$\min_{I^0,E,\tau} ||I^0||_* + \lambda_1||E||_1 + f_{align}(\tau, \mathcal{W}), \text{s.t.} I \circ \tau = I^0 + E \tag{1}$$

where $||I^0||_* + \lambda_1||E||_1$ is a term reflecting the glyph property of rectified image which is adopted from the low-rank transform approach [16]. To be precise, when an input image I is transformed by the transform parameter τ, it is decomposed into a low-rank matrix I^0 and a sparse error matrix E (refer to [16] for details of these terms). However, the optimization of the low-rank term sometimes yields misaligned results (as shown in Fig. 1), and hence we attempt to improve this case by introducing the last alignment term $f_{align}(\tau, \mathcal{W})$.

B. Alignment term

For using the above-stated properties, we need to segment the characters and estimate the alignment parameters though not complete. In this subsection, we assume that we have tentative segmentation result and alignment parameters (in the process of iteration), and how to segment the text and how to obtain the alignment parameters will be explained later.

With the given segments and alignment parameters, the alignment term $f_{align}(\tau, \mathcal{W})$ is designed as

$$f_{align}(\tau, \mathcal{W}) = \lambda_2 f_{vert}(\tau, \mathcal{W}) + \lambda_3 f_{hori}(\tau), \tag{2}$$

where $f_{vert}(\tau, \mathcal{W})$ reflects the vertical alignment, and $f_{hori}(\tau)$ reflects the horizontal alignment of characters.

For explaining the terms in detail, let us first denote a set of pixels that correspond to the i-th segment as $C_i(\tau)$, when the text image is transformed by the given parameter τ. For each segment C_i (which may not exactly correspond to a character, but actually part of character or overlap of characters), its top and bottom vertical positions after the transformation τ are denoted as $T_i(\tau)$ and $B_i(\tau)$, respectively:

$$T_i(\tau) = \max_{j \in C_i(\tau)} [p_j * \tau]_y, \tag{3}$$

$$B_i(\tau) = \min_{j \in C_i(\tau)} [p_j * \tau]_y, \tag{4}$$

where j is a pixel index of the i-th segment, $p * \tau$ is the transformation of a point p by τ, and $[p]_y$ means the vertical

Algorithm 1: Overall iterative algorithm

Data: Input image I, initial transformation parameters τ

Result: Solutions I^{0*}, E^*, τ^*

while do

  Step 1 : Determine the number of character N and character segments results $\{C_i(\tau)\}_{i=1}^{N}$

  Step 2 : Estimate the alignment parameters $\mathcal{W}$

  Step 3 : Estimate the rectification variables (I^0, E, τ) by minimizing the cost function (1)

end

position of p. By using the alignment parameters and above two equations, we represent the vertical alignment term as

$$f_{vert}(\tau, \mathcal{W}) = \sum_{i=1}^{N} \omega_i \left(T_i(\tau) - y_t\right)^2 + \sum_{i=1}^{N} \mu_i \left(B_i(\tau) - y_b\right)^2, \tag{5}$$

where N is the number of segments, and y_t and y_b are the vertical positions of top and bottom lines in Fig. 2 respectively.

Then, let us denote the width of the i-th character after the transformation τ as $W_i(\tau)$:

$$W_i(\tau) = \max_{j \in C_i(\tau)} [p_j * \tau]_x - \min_{j \in C_i(\tau)} [p_j * \tau]_x, \tag{6}$$

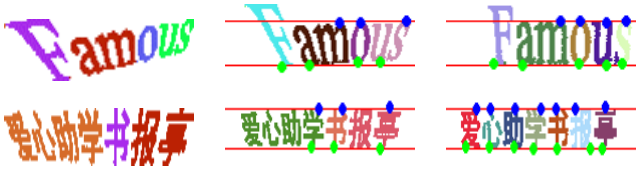
where $[p]_x$ means the horizontal position of p. By using the above equation, we represent the horizontal alignment term as

$$f_{hori}(\tau) = \sum_{i=1}^{N} W_i(\tau) \tag{7}$$

which encodes the idea that the horizontal alignment term has small value when the character segments are well aligned as stated previously. The problem with our cost function is that it is non-convex and involves many (continuous and discrete) variables. Therefore, its optimization is a difficult problem and we discuss its optimization method in the next section.

III. Overall Algorithm

For defining the alignment term in (1), we need the character segmentation of the given text. However, exact character segmentation is a difficult problem especially in the case of distorted images. Since better segmentation needs better rectification and vice versa, we solve this problem by performing the segmentation (along with the alignment parameter estimation) and rectification iteratively. Fig. 3 shows the results of some steps of our algorithm, where we can see that the initial segmentation is not complete at first, but the segmentation is improved along with the rectification as the algorithm is iterated. For our iterative scheme, we first perform the character segmentation in the rectified image (by the current estimated τ) using a projection profile method. After the character segmentation, we estimate the alignment parameters by using the least squares line fitting with RANSAC. Then, we perform rectification by minimizing the cost function in (1). For minimizing this non-convex function, we add auxiliary



(a) (b) (c)

Fig. 3: Our iterative rectification and segmentation scheme. (a) Input and the initial character segments. The same colored pixels mean that they are grouped in a segment. Specifically, the upper row text is initially segmented into four segments and the lower one into three. (b) Rectified images and their character segments after the first iteration. (c) Rectified images and their character segments after the second iteration. Top (blue) points and Bottom (green) points are aligned to two horizontal (red) lines, and characters are better rectified and segmented as the algorithm is iterated.

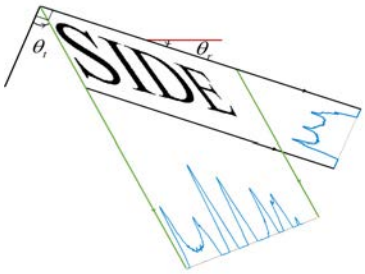


Fig. 4: Estimation of the skew and shearing angles using the projection profile analysis.

variables and solve the linearized problem iteratively. The whole optimization process is summarized in Algorithm 1, and the explanation is detailed in the following subsections.

A. Initialization

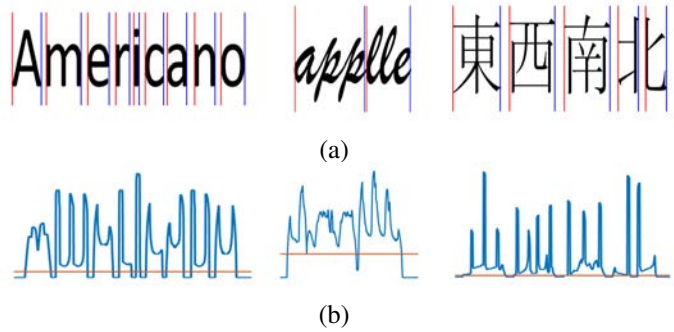
For the initialization of transformation parameters τ, we estimate rough skew and shearing angles. First, for the skew estimation, we compute the projection profiles and consider the angle yielding the most compact profile as the skew angle θ_r. After correcting the skew with the estimated angle, we estimate the shearing angle θ_t by maximizing the length sum of zero-runs in horizontal profiles (i.e., intervals between super-pixels) in the search range of $(-\pi/3, \pi/3)$. Then, the initial homography matrix is given by

$$\begin{pmatrix} 1 & -\tan\theta_t & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta_r & -\sin\theta_r & 0 \\ \sin\theta_r & \cos\theta_r & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{8}$$

and a transformation parameter vector τ is initialized by this homography. This step is illustrated in Fig. 4.

B. Character segmentation

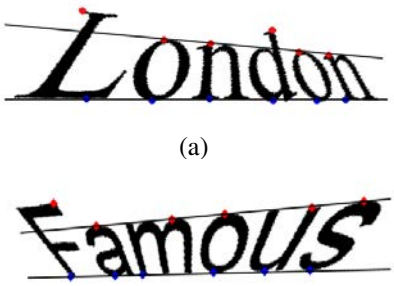
In order to encode the "alignment of characters" into our cost function, we need to separate the text into individual



(a)

(b)

Fig. 5: Results of character segmentation. (a) Character segments. (b) (Blue) graph and horizontal (red) lines mean projection profiles and mean stroke width values, respectively.



(a)

(b)

Fig. 6: Results of line fitting on computed top and bottom points with RANSAC.

characters or at least into some segments. First, we extract connected components (CCs) and estimate the mean stroke width of the connected components corresponding to scene text. Intuitively, the stroke width represents how far the (imaginary) pen moves to write a given CC. For this representation, we estimate the stroke widths of all CCs [25] and calculate the mean stroke width. Second, we compute the horizontal projection profile (lower blue graph in Fig. 4) and separate the characters by comparing the projection profile width and mean stroke width value. This is illustrated in Fig. 5, where the red line is the mean stroke value, and the text is segmented when the projection profile is larger than the mean stroke value. As shown in the figure, English and Chinese text are separated by this projection profile method quite well. Although Chinese characters are sometimes over-segmented as shown in Fig. 5(a) (there are actually four characters where the last character is segmented into two), the exact segmentation is not necessary in our case as long as we can find upper and lower point of the segment for encoding the alignment properties. In the case of cursive or slanted writing, it is prone to under-segmentation as shown in Fig. 5(a) (apple in the middle), but we can still find upper and lower points that help to align the text. As defined previously, the pixels in the i-th segment are the elements of $C_i(\tau)$.

C. Estimation of the alignment parameters

As defined in Subsectin 3.1, the alignment parameters $\mathcal{W} = \{(\omega_i, \mu_i)\}_{i=1}^N$ are defined for determining whether the i-th segment's top and/or bottom points are aligned to two lines l_t and l_b respectively. These two lines are parallel in the case of well-rectified image as shown in Fig. 2, but they are actually not when the current estimated τ is still not perfectly optimized. However, since a line is transformed to a line by a projective transform, we find (slanted) lines that fit the top/bottom points of the segments as illustrated in Fig. 6. The line fitting is performed by the least squares fitting of the computed top and bottom points with RANSAC. The results are also illustrated in Fig. 6, which means that ω_i for $i = 2, 3, 5, 6$ are 1, and $\omega_1 = \omega_4 = 0$ for Fig. 6(a), and $\omega_1 = 0$ and $\omega_i = 1$ for the rest of i in the case of Fig. 6(b) (Note that top points of L, d, and F in the figure are determined to be outliers). In both cases, $\mu_i = 1$ for all i as all the segments are determined to touch the bottom line.

D. Cost function optimization for rectification

We estimate the variables $(I^0,\ E,\ \tau)$ by minimizing the proposed cost function in (1), which is non-convex and thus difficult to optimize. Therefore, we first transform the cost function into a convex function by introducing auxiliary variables $\Gamma = \{(\alpha_i, \beta_i, \gamma_i)\}_{i=1}^N$:

$$\begin{aligned}\min_{I^0,E,\tau,\Gamma} ||I^0||_* \quad &+\lambda_1||E||_1 + \lambda_2 \sum_{i=1}^{N} \omega_i(\alpha_i - y_t)^2 \\ + \quad &\lambda_2 \sum_{i=1}^{N} \mu_i(\beta_i - y_b)^2 + \lambda_3 \sum_{i=1}^{N} \gamma_i, \\ &\text{s.t.} I \circ \tau = I^0 + E, \\ &T_i(\tau) = \alpha_i, \\ &B_i(\tau) = \beta_i, \\ &W_i(\tau) = \gamma_i.\end{aligned} \tag{9}$$

Then, we linearize the constraints around the current estimated value τ and solve the linearized problem [16], which is written as

$$\begin{aligned}\min_{I^0,E,\Delta\tau,\Gamma} ||I^0||_* + \lambda_1||E||_1 + \lambda_2 \sum_{i=1}^{N} \omega_i(\alpha_i - y_1)^2 \\ +\lambda_2 \sum_{i=1}^{N} \mu_i(\beta_i - y_b)^2 + \lambda_3 \sum_{i=1}^{N} \gamma_i, \\ \text{s.t.} \nabla I \Delta\tau + I \circ \tau = I^0 + E, \\ \nabla T_i \Delta\tau + \max_{j \in C_i(\tau)} [p_j * \tau]_y = \alpha_i, \\ \nabla B_i \Delta\tau + \min_{j \in C_i(\tau)} [p_j * \tau]_y = \beta_i, \\ \nabla W_i \Delta\tau + \max_{j \in C_i(\tau)} [p_j * \tau]_x - \min_{j \in C_i(\tau)} [p_j * \tau]_x = \gamma_i,\end{aligned} \tag{10}$$

where $\nabla I = \frac{\partial(I \circ \tau)}{\partial \tau}$, $\nabla T_i = \frac{\partial(T_i(\tau))}{\partial \tau}$, $\nabla B_i = \frac{\partial(B_i(\tau))}{\partial \tau}$, and $\nabla W_i = \frac{\partial(W_i(\tau))}{\partial \tau}$. To solve this linearized convex problem, we use the Augmented Lagrange Multiplier (ALM) method [26].

Algorithm 2: Solving the problem (9)

Data: Input image I, initial transformation parameters τ, the number of characters N, character segments $\{C_i\}_{i=1}^{N}$, alignment parameters $\mathcal{W} = \{(\omega_i, \mu_i)\}_{i=1}^{N}$.
Result: Solution I^{0^*}, E^*, τ^*
while *not converged* **do**
  Step 1 : Compute Jacobians ∇I, ∇T_i, ∇B_i, and ∇W_i
  Step 2 : Solve the linearized problem in (10)
  Step 3 : Update the transformation: $\tau \leftarrow \tau + \Delta\tau$
end

However, since (10) is a local approximation of the original non-convex problem, we find the solution to the original problem (9) by solving the linearized problem iteratively. This iterative solver is summarized in Algorithm 2.

IV. Experimental results

We evaluate our method on the scene text dataset [20], which is consisted of real and synthetic text images. The real images are from MSRA-TD 500 dataset [2] and synthetic images are obtained by applying homography transformation to ICDAR 2013 Robust Reading Competition dataset [27], which includes English and Chinese characters. In the experiments, we resize the input so that their pixel-resolutions become 3,500 pixels. We set $\lambda_1 = 20/\min(w,h)$ and $\lambda_2 = \lambda_3 = 20/N$, where N is the number of character segments, and w, h are the image width and height. Also, we set $y_t = 0$ and $y_b = h$. Our algorithm is implemented with C++, which takes $2 \sim 5$ seconds for the rectification of an image (3500-pixel resolution) on Intel(R) i5(TM) CPU(3.40 GHz), and its running time is approximately 3 times larger than the conventional method [16].

For the objective evaluation of the rectification, we compare our method with the conventional method [16]. The performance is evaluated in terms of OCR accuracy, which is defined as

$$\text{accuracy}(R,G) = 1 - \frac{L(R,G)}{\max(\#R, \#G)}, \qquad (11)$$

where R is a recognized string, G is the ground truth string, $\#(\cdot)$ is the number of characters in the string, and $L(x,y)$ means the Levenshtein distance between two strings [28]. The distance is defined as the minimum number of character edits (insertion, deletion, and substitution) to transform a wrong string to the correct one. For the OCR, we use the Tesseract OCR engine [1].

Experimental results for OCR accuracy are summarized in Table I and Table II. It can be seen that the proposed method shows higher accuracy for both real and synthetic text images. Some rectified results are shown in Figs. 7 and 8 for visual comparison. We can see that the conventional method has difficulties in the case of severe distortions, whereas the proposed algorithm works for these cases. Full-resolution of more inputs/results can be found at: https://github.com/xellows1305/Scene-Text-Rectification, where we will also make the source or executable codes publicly available.

TABLE I: OCR accuracy of the proposed and conventional method [16] on real scene text images.

OCR accuracy	English	Chinese
Input image	0.6128	0.4566
TILT [16]	0.9178	0.6326
Proposed	**0.9421**	**0.7183**

TABLE II: OCR accuracy of the proposed and conventional method [16] on synthetic text images.

OCR accuracy	English	Chinese
Input image	0.1475	0.1178
TILT [16]	0.7974	0.4005
Proposed	**0.9023**	**0.6531**

V. Conclusion

We have proposed a new scene text rectification algorithm, which adds an additional constraint to the conventional low-rank term. We define the character alignment constraint that the characters in the text touch two virtual parallel (top/bottom) lines. We perform rough segmentation using projection profile method at first, and finding the transform parameter from the touching lines and the text segmentation are iterated for progressively improving the results. From this process, we could find a projective transform that turns two lines into parallel ones. Experimental results on natural and synthetic images show that the OCR accuracy of the proposed algorithm is higher than the conventional method based on the low-rank property. As a future work, we will incorporate the dewarping into the deep-learning based OCR methods such as recent LSTM-based of Tesseract.

Acknowledgement

This research was supported by the Ministry of Trade, Industry, and Energy (MOTIE), Korea, under the "Regional Specialized Industry Development Program (R&D, P0002072)" supervised by the Korea Institute for Advancement of Technology (KIAT).

References

[1] R. Smith, "An overview of the tesseract ocr engine," in *IEEE International Conference on Document Analysis and Recognition*, vol. 2, 2007, pp. 629–633.
[2] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1083–1090.
[3] H. I. Koo and D. H. Kim, "Scene text detection via connected component clustering and nontext filtering," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2296–2305, 2013.
[4] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "Photoocr: Reading text in uncontrolled conditions," in *IEEE International Conference on Computer Vision*, 2013, pp. 785–792.
[5] S. N. Srihari and V. Govindaraju, "Analysis of textual images using the hough transform," *Machine vision and Applications*, vol. 2, no. 3, pp. 141–153, 1989.
[6] C. Sun and D. Si, "Skew and slant correction for document images using gradient direction," in *IEEE International Conference on Document Analysis and Recognition*, vol. 1, 1997, pp. 142–146.

(a) (b) (c) (d) (e) (f)

Fig. 7: Rectification results on real scene text images. (a), (d) input distorted image, (b), (e) the conventional method [16], (c), (f) the proposed method.

(a) (b) (c) (d) (e) (f)

Fig. 8: Rectification results on synthetic text images. (a), (d) input distorted image, (b), (e) the conventional method [16], (c), (f) the proposed method.

[7] P. Clark and M. Mirmehdi, "Rectifying perspective views of text in 3d scenes using vanishing points," *Pattern Recognition*, vol. 36, no. 11, pp. 2673–2686, 2003.

[8] I. Bar-Yosef, N. Hagbi, K. Kedem, and I. Dinstein, "Fast and accurate skew estimation based on distance transform," in *IEEE IAPR International Workshop on Document Analysis Systems*, 2008, pp. 402–407.

[9] J. Liang, D. DeMenthon, and D. Doermann, "Geometric rectification of camera-captured document images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 591–605, 2008.

[10] H. I. Koo, J. Kim, and N. I. Cho, "Composition of a dewarped and enhanced document image from two view images," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1551–1562, 2009.

[11] B. Epshtein, "Determining document skew using inter-line spaces," in *IEEE International Conference on Document Analysis and Recognition*, 2011, pp. 27–31.

[12] B. S. Kim, H. I. Koo, and N. I. Cho, "Document dewarping via text-line based optimization," *Pattern Recognition*, vol. 48, no. 11, pp. 3600–3614, 2015.

[13] S. Messelodi and C. M. Modena, "Automatic identification and skew estimation of text lines in real scene images," *Pattern Recognition*, vol. 32, no. 5, pp. 791–810, 1999.

[14] G. K. Myers, R. C. Bolles, Q.-T. Luong, J. A. Herson, and H. B. Aradhye, "Rectification and recognition of text in 3-d scenes," *International Journal of Document Analysis and Recognition*, vol. 7, no. 2-3, pp. 147–158, 2005.

[15] M. Bušta, T. Drtina, D. Helekal, L. Neumann, and J. Matas, "Efficient character skew rectification in scene text images," in *Springer Asian Conference on Computer Vision*, 2014, pp. 134–146.

[16] Z. Zhang, X. Liang, A. Ganesh, and Y. Ma, "TILT: transform invariant low-rank textures," in *Springer Asian Conference on Computer Vision*, 2010, pp. 314–328.

[17] X. Zhang, Z. Lin, F. Sun, and Y. Ma, "Rectification of optical characters as transform invariant low-rank textures," in *IEEE International Conference on Document Analysis and Recognition*, 2013, pp. 393–397.

[18] ——, "Transform invariant text extraction," *The Visual Computer*, vol. 30, no. 4, pp. 401–415, 2014.

[19] C. Merino-Gracia, M. Mirmehdi, J. Sigut, and J. L. González-Mora, "Fast perspective recovery of text in natural scenes," *Image and Vision Computing*, vol. 31, no. 10, pp. 714–724, 2013.

[20] B. Wang, C. Liu, and X. Ding, "A scheme for automatic text rectification in real scene images," in *SPIE/IS&T Electronic Imaging*, 2015, pp. 94 080M–94 080M.

[21] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, "Robust scene text recognition with automatic rectification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4168–4176.

[22] Z. Cheng, F. Bai, Y. Xi, and G. Zheng, "Focusing attention: Towards accurate text recognition in natural images," in *IEEE Internatioinal Conference on Computer Vision*, 2017, pp. 5086–5094.

[23] Y. Lu, "Machine printed character segmentation?; an overview," *Pattern Recognition*, vol. 28, no. 1, pp. 67–80, 1995.

[24] R. G. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 7, pp. 690–706, 1996.

[25] J. Ryu, H. I. Koo, and N. I. Cho, "Language-independent text-line extraction algorithm for handwritten documents," *IEEE Signal Processing Letters*, vol. 21, no. 9, pp. 1115–1119, 2014.

[26] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[27] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. de las Heras, "ICDAR robust reading competition," in *IEEE International Conference on Document Analysis and Recognition*, 2013, pp. 1484–1493.

[28] V. Levenstein, "Binary codes capable of correcting spurious insertions and deletions of ones," *Problems of Information Transmission*, vol. 1, no. 1, pp. 8–17, 1965.