Fast Perspective Recovery of Text in Natural Scenes

Carlos Merino-Gracia, Majid Mirmehdi, José Sigut, José L. González-Mora

PII: S0262-8856(13)00106-6
DOI: doi: 10.1016/j.imavis.2013.07.002
Reference: IMAVIS 3249

To appear in: *Image and Vision Computing*

Received date: 18 July 2012
Revised date: 13 June 2013
Accepted date: 18 July 2013

Fast Perspective Recovery of Text in Natural Scenes[☆]

Carlos Merino-Gracia[a,b], Majid Mirmehdi[b], José Sigut[c], José L. González-Mora[a]

[a] Neurochemistry and Neuroimaging Laboratory. University of La Laguna. Spain.
[b] Visual Information Laboratory. University of Bristol. United Kingdom.
[c] Department of Systems Engineering and Control and Computer Architecture. University of La Laguna. Spain.

Abstract

Cheap, ubiquitous, high-resolution digital cameras have led to opportunities that demand camera-based text understanding, such as wearable computing or assistive technology. Perspective distortion is one of the main challenges for text recognition in camera captured images since the camera may often not have a fronto-parallel view of the text. We present a method for perspective recovery of text in natural scenes, where text can appear as isolated words, short sentences or small paragraphs (as found on posters, billboards, shop and street signs etc.). It relies on the geometry of the characters themselves to estimate a rectifying homography for every line of text, irrespective of the view of the text over a large range of orientations. The horizontal perspective foreshortening is corrected by fitting two lines to the top and bottom of the text, while the vertical perspective foreshortening and shearing are estimated by performing a linear regression on the shear variation of the individual characters within the text line. The proposed method is efficient and fast. We present comparative results with improved recognition accuracy against the current state-of-the-art.

Keywords: scene text extraction, perspective recovery, homography rectification

1. Introduction

Efficient and fast comprehension of text in our environment is an important aspect of scene understanding for a variety of application areas, e.g. for automatic and assisted navigation of robots and humans respectively [1–4]. Images from a mobile camera, indoors or outdoors, pose considerable challenges to text understanding, such as blurred or out of focus frames, uneven lighting, complex backgrounds, and lens distortion. One of the main issues is perspective distortion as the camera may not necessarily have a fronto-parallel view of the text. There have been rare attempts to directly recognise characters with perspective deformation, e.g. [5], but in general, even when the region of text can be delineated reasonably well, the accuracy of OCR engines degrades quickly with increasing perspective effects.

The focus of this work then is on perspective recovery of text in natural scenes. Our aim is to obtain a fronto-parallel reconstruction of an image patch with scene text that improves the text recognition accuracy by off-the-shelf OCR software. The characteristics of scene text are fundamentally different from those of document images with text appearing in various orientations including differing orientations within the same image. Such texts usually appear in the form of isolated words or short sentences in diverse typefaces.

We wish to recover text (e.g. on posters, billboards, shops and street signs) from images with enough resolution to make the segmentation of individual characters possible. We expect only a single frame – so no temporal information – and no camera parameters, e.g. the focal length would be unknown. The 3D orientation of the text should not matter (except for extreme views), and the only assumption we make is that the text is laid out in a straight line on a planar surface.

The method proposed here computes a rectifying homography to reconstruct a fronto-parallel image for a line of text that may have been affected by perspective transformations, such as horizontal perspective foreshortening, shearing, and vertical perspective foreshortening. We correct horizontal perspective foreshortening by fitting two lines to the top and bottom of the text. The shearing and vertical perspective foreshortening are rectified by first estimating a shearing value for each character to then perform a linear regression on the shear variation across the text line.

Our experimental results include a systematic test of texts, obtained from the ICDAR 2011 Robust Reading Competition datasets [6, 7], synthetically regenerated at various orientations to establish a groundtruth for performance comparison, followed by results on natural scene images. We present performance evaluation showing significant increase in recognition accuracy, across a wide range of viewing angles, compared against the unrectified image

[☆] This work was carried out at Bristol University by Carlos Merino-Gracia, who is funded by the Spanish Ministerio de Industria y Comercio (project TSI-020100-2010-346).

Email addresses: cmerino@ull.es (Carlos Merino-Gracia), majid@cs.bris.ac.uk (Majid Mirmehdi), sigut@isaatc.ull.es (José Sigut), jlgonzal@ull.es (José L. González-Mora)

and the scene text perspective recovery technique by Myers et al. [8].

Next, in Section 2 the problem is formally defined and set in context with respect to related works, followed by a detailed description of our proposed method in Section 3. Experimental results are presented in Section 4. We conclude our work in Section 5 and point to future directions.

2. Problem statement and related work

Before the emergence of camera based document acquisition, in-plane rotation or *skew* was the main geometric correction that document analysis systems had to deal with. Extensive literature exists in the area of document skew estimation, for example for some surveys see [9–11].

For camera based rectification of text, there are more degrees of freedom to consider. Assuming text lies on a planar surface, the process of perspective recovery of text can be modelled as a projective transformation [12] between the source image and a target image. As the projective transformation preserves linearities, a rectangle enclosing the text in its original plane and orientation is seen as a quadrilateral in the source image and would need to be mapped to a rectangle in the target image (see Fig. 1). This projective transformation or homography is represented by a 3 × 3 mapping matrix:

$$\mathbf{p}' = \mathbf{H}\,\mathbf{p}\,, \qquad (1)$$

where $\mathbf{p} = [\,\mathrm{x}\ \mathrm{y}\ 1\,]^\intercal$ and $\mathbf{p}' = [\,c\mathrm{x}'\ c\mathrm{y}'\ c\,]^\intercal$ are homogeneous coordinate points in the source and target images respectively and $\mathbf{H}$ is the homography matrix.

The homography has 8 degrees of freedom which can be decomposed into: translation and scale along each axis, Euclidean rotation, shear and two perspective foreshortenings along each axis respectively. As pointed out by Myers et al. [8], some of the degrees of freedom affect recognition more than others: OCR engines can deal with translation and scaling well, and rotation (or *skew*) is also handled by current OCR systems (albeit for a limited range of angles). Therefore, OCR-wise, the problem can be reformulated as correcting the distortions produced by shear and the two perspective foreshortenings, or alternatively, as estimating the location of two vanishing points within the image plane.

Pilu [13] and Clark and Mirmehdi [14, 15] were among the first to look at perspective recovery of camera acquired *document images*. Pilu [13] looked at the high level organization of text within documents as a basis for extracting illusory visual clues and computing the vanishing points to perform rectification. He employed a saliency measure between text connected components to form lines of text and estimate the horizontal vanishing point. Then, he used a set of carefully chosen rules of association between components in different lines to construct a set of candidate vertical lines which defined the vertical vanishing point. However, given that vertical clues are more scarce and difficult to get, Pilu [13] acknowledged that his vertical vanishing point estimation is not as reliable as the horizontal one. Clark and Mirmehdi [14] proposed two distinct perspective correction techniques based on extracting cues from higher level structures of text within document images. In their first technique, they searched for quadrilaterals in the image that would enclose text (e.g. paper borders, frames) and use it to compute the projective transformation. In their second and more complex approach, they considered the text itself only to infer the two vanishing points. The horizontal vanishing point was estimated by computing a projection profile for every possible vanishing point in a 2D polar search space around the image centre. Then, the vertical vanishing point was obtained by projecting lines from the left and right margin lines, which restricts this technique to fully justified paragraphs. This process was later refined [15] to include left or right-only justified paragraphs by employing the spacing between lines in the computation of the vertical vanishing point. In a similar fashion to the first technique of [14], Cambra and Murillo [16] also looked for borders enclosing text regions for rectification and implemented it on a mobile phone.

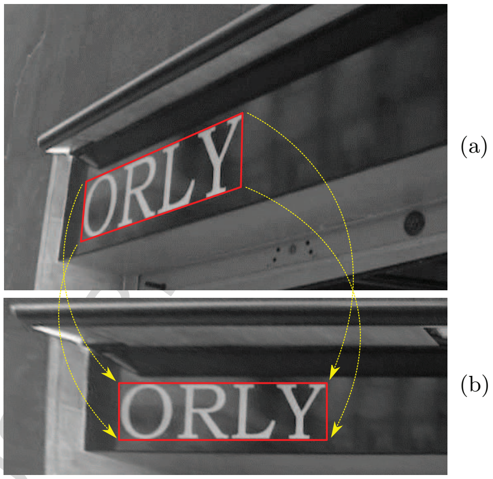


Figure 1: *A projective transformation of text.* A rectangle enclosing the text is seen as a quadrilateral in the *source image* (a) and is mapped into a rectangle in the *target image* (b).

More recent works focused on document images include Stamatopoulos et al. [17] and Liang et al. [18] where perspective recovery was considered along with dewarping. In [17], after a word and line detection stage, text was rectified in two steps: a coarse correction to remove the global distortions of the image and a fine correction to restore the local deformations. The coarse rectification proceeds by warping an area delimited by two curves fitted to the top and bottom text lines of the document, along with the left and right boundaries of the text. Another text detection stage precedes the fine rectification step, in which a baseline is fitted to every word and used to rotate and translate each of them independently in the output image. Liang et al. [18] used the notion of *texture flow*, where certain patterns in textured surfaces can give a sense of continuous flow. Two texture flow orientations (named major and minor) were found in document images, aligned with the directions of the text line and the vertical strokes respectively. The major texture flow was determined by applying projection profiles locally, where directional filters were used to obtain the the minor texture flow. The method differentiates between flat and curved document images, the latter involving not only rectification, but document dewarping. In the case of flat documents, the lines projected by the two texture flow directions converge into vanishing points that were then used to compute the rectification.

The methods described above cannot be applied to scene text, since, to find orientation cues, they rely on how text is structured and organized within documents, i.e. as groups of lines. The most relevant work, specifically dealing with 3D scene text recovery, is by Myers et al. [8] whose method deals with individual or isolated text lines found in everyday scenes, particularly outdoors. In that work, images are first segmented and individual lines of text are extracted. The text lines are rotated at various angle increments and horizontal projection profiles for each angle are computed. By measuring the slope on the sides of the projection profile, top and bottom angles can be estimated, allowing for the estimation of the horizontal vanishing point and a *partial* rectification of the text by removing the horizontal foreshortening.

As expressed earlier, correcting shear and vertical foreshortening is a challenging problem due to the difficulty of obtaining accurate vertical cues for text – even more so when only one text line is being considered. Myers et al.'s [8] view of this is that a weak perspective deformation is expected in the vertical axis on natural scenes, as cameras are usually oriented closely to the horizontal and, in the real-world, text is laid out on vertical surfaces. Therefore, assuming that the vertical vanishing point lies at infinity, they estimate a single shear angle for the whole line by also employing vertical projection profiles. However, they also acknowledge that, when the perspective distortion is significant, their method of correcting shear produces (after rectification) a line of text where the vertical strokes vary in angle with respect to their horizontal position. This is more apparent when images obtained with hand-held or wearable cameras are considered, since the camera could be pointing to text at more extreme orientations. Furthermore, in Myers et al. [8], a large number of possible shear angles within an interval have to be evaluated, which involves a whole image transformation and the computation of a projection profile for each angle. This makes their method inefficient, or if the number of evaluated angles is reduced, inaccurate.

Several systems have been proposed where only an affine rectification of text was performed. In the work by Chen et al. [19], text was segmented using an edge detector combined with a Gaussian mixture model (GMM) to separate the text from the background. Characters were grouped together by means of a similarity function and a Hough transform on the character's centre points was used to detect linear distribution patterns. A minimum area rectangle was then fitted around each character, aligned with the main direction of the character's group. The most salient rectangle of each group –selected as the one with maximum average edge intensity inside the rectangle– was used to compute two (top and bottom) parallel lines for the group. Two additional parallel lines were also computed using the left- and right-most character rectangles. With these lines an affine rectification of the text group was then computed. Yamaguchi et al. [20] employed a two step rectification for recognising *digits* in natural scenes. They made the assumption that the text plane is close to a fronto-parallel view from the camera, thus considering that the vanishing points are far away or close to infinity. Under these conditions, the text was rectified by applying two affine transformations in sequence: one to remove the skew (or in-plane rotation) and a second to remove the slant (or *shear*). The skew angle was obtained with a modified Hough transform on the centre points of each character. Then, rotated minimum area rectangles were circumscribed to each character to obtain an average slant angle for the whole line. Therefore, as a true projective transformation was not being performed on either of these methods, they will also produce incorrect rectifications when significant perspective distortions are in play.

A completely different approach was employed by Li and Tan [5] by recognising characters with perspective distortion directly. This technique was aimed at recognising symbols (e.g. single characters, traffic signs, logos). For this purpose they introduced an image descriptor which is invariant to projective transformations. The authors demonstrated the increased recognition accuracy of their method over state-of-the-art image matching algorithms for symbols with severe perspective deformations. However, when considered for scene text recognition, this approach lacks all the technical advances that current OCR engines apply besides character recognition: joining or splitting of components to form characters, text line and word formation, statistical dictionary search, etc. If these techniques were to be adapted and applied directly on the unrectified image, they would certainly benefit from having an estimation of the true orientation of the text line.

In this work we perform a full perspective rectification of the text, relying only on the geometry of the characters themselves. We do not assume the vertical vanishing point to lie at infinity, thus allowing for bigger variations of shear within the line of text, and our method deals efficiently and accurately with a larger range of view angles.

3. Proposed Method

Our full scene text extraction system comprises several stages: text detection, text grouping and orientation detection. Since the focus of our work here is on the latter two stages, which encompass our introduction and evaluation of the proposed perspective rectification method, we only

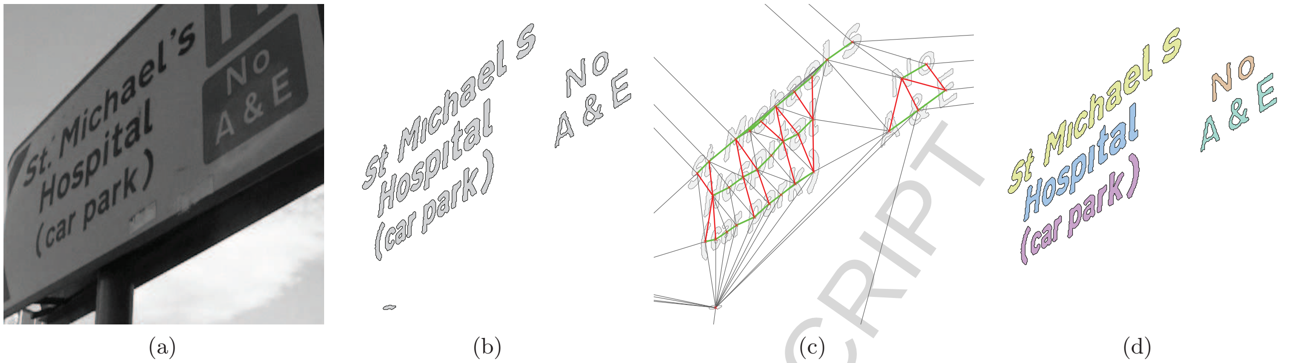

(a) (b) (c) (d)

Figure 2: The result of the segmentation and grouping steps: (a) the original image, (b) the segmented components, (c) the association graph (grey edges were removed during saliency filtering and red edges were removed during histogram filtering; the green edges represent the segmented text lines), and (d) the grouped text lines.

briefly review our initial detection stage to set the scene. More details of our various text detection methods can be found in [1, 21]. Any other text detection technique, such as [22] or [23], which output candidate regions of text can also be used as input to our perspective recovery method.

3.1. Text detection

The input image has to be segmented to obtain a set of regions representing individual characters or small groups of them. For this purpose, a text detector which we previously developed [21] is employed. A brief outline of the algorithm follows.

Adaptive thresholding is applied to binarize the input image and retrieve a set of connected component regions (CCs). A tree is then constructed to represent the topological relationship between these CCs. A key step of this algorithm is the hierarchical filtering of the tree nodes, based on the assumption that on real-world images with scene text, structural elements (such as sign borders, posters frames etc.) can be discarded purely based on their hierarchical relationship with other text regions. Additionally, the tree filtering approach allows for the segmentation of dark and bright text in one pass only. The CCs are then classified by means of a cascade of text filters that operate on characteristics such as size and contrast against the background, and an eigenvector based texture measure adapted from [24]. Figure 2a shows an example image and Fig. 2b illustrates the corresponding CCs (or regions) detected at this stage.

3.2. Text grouping

The CC regions detected above will be a fragmented representation of the characters in words, and words in short sentences. We need to group these together to reform words and sentences, in order to be in a position to extract common clues for the perspective recovery of the text. This reformation is performed by first determining which CC regions are connected by evaluating a visual saliency measure between each pair of regions, and then by searching for dominant orientations to separate independent lines of text.

Saliency filtering – First, a Delaunay triangulation [25] joining the centre points of every CC is performed, with the centre points being the centre of mass of each region. The Delaunay triangulation enables us to efficiently construct a neighbour relationship graph between all the components. Figure 2c shows the result of the Delaunay triangulation. For every edge of the resulting graph, which represents a pair of adjacent CCs, a saliency measure is computed.

We use the two saliency operators introduced by Pilu [13]: the *blob dimension ratio* (BDR; γ) and the *relative minimum distance* (RMD; λ). Given two CC regions, $\mathcal{A}$ and $\mathcal{B}$, BDR evaluates the similarity in size between them, i.e.

$$\gamma(\mathcal{A},\mathcal{B}) = \frac{\mathcal{A}_{\min} + \mathcal{A}_{\max}}{\mathcal{B}_{\min} + \mathcal{B}_{\max}} , \qquad (2)$$

where $\mathcal{A}_{\min}$, $\mathcal{B}_{\min}$, $\mathcal{A}_{\max}$ and $\mathcal{B}_{\max}$ are the minimum and maximum axes of regions $\mathcal{A}$ and $\mathcal{B}$ respectively, while RMD evaluates the distance of the two CCs relative to their respective sizes, i.e.

$$\lambda(\mathcal{A},\mathcal{B}) = \frac{D_{\min}}{\mathcal{A}_{\min} + \mathcal{B}_{\min}} , \qquad (3)$$

where $D_{\min}$ is the minimum distance between two regions. The minimum and maximum axes are extracted from the minimum enclosing box (rotated rectangle) around the regions. The combined saliency operator between the two text regions is then:

$$\mathbf{P}(\mathcal{A},\mathcal{B}) = N(\lambda(\mathcal{A},\mathcal{B}),1,2) \cdot N(\gamma(\mathcal{A},\mathcal{B}),0,4) , \qquad (4)$$

where $N(x,\mu,\sigma)$ is a Gaussian distribution with mean μ and standard deviation σ (the parameters were determined experimentally by Pilu [13]). Edges with $\mathbf{P}(\mathcal{A},\mathcal{B}) < 0.9$ are removed from the graph. In Fig. 2c, edges removed during the saliency filtering are represented in grey.

Histogram filtering – After the saliency filtering, every remaining connected subgraph is a candidate text group,

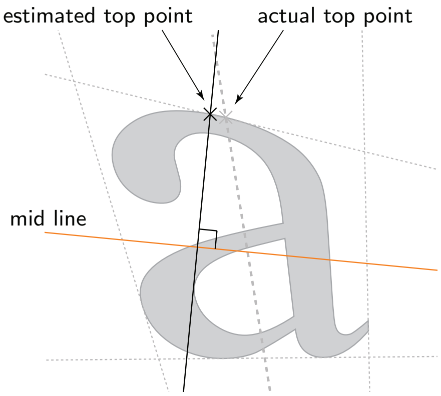


Figure 3: Top point estimation – on severely perspective distorted characters the estimated top point and the actual top point might not correspond.

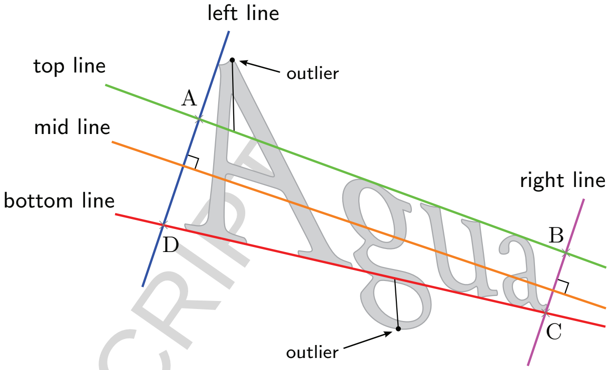


Figure 4: Top, mid, bottom, left and right lines along with the formed quadrilateral. The outliers of the line estimation are also illustrated here.

each of them possibly containing one or more lines of text. The text groups are then evaluated to find the dominant orientation and to separate the individual text lines.

The angle between each edge of the subgraph and the x-axis is computed and reduced to the $[0, \pi)$ interval. This angle interval is divided into 8 bins and then a histogram of angle distribution of the graph edges is built. The histogram bin containing the highest number of edges is selected. Every edge that does not belong to that bin or to any of its two adjacent bins is removed from the graph. The remaining edges belong to the dominant orientation of the text line. After the removal of these graph edges, the original subgraph may be split into smaller subgraphs since the original candidate text groups might have had multiple text lines that are now separated. In Fig. 2c, filtered edges at this stage are represented in red, and the remaining connected subgraphs are represented in green. Finally, Fig. 2d shows the result of the text segmentation and grouping, in which each segmented text line is drawn in a different color.

Now every remaining connected subgraph contains only one text line. A text line is defined by a set of N characters $\mathcal{C}_i$, $i = 1, \ldots N$, each character being a connected component. The perspective estimation relies on the character's contour points and, for efficiency reasons, the convex hull of each CC is used as the contour for the character. For the remainder of this paper, the unit of work is the text line.

3.3. Orientation detection

The orientation estimation technique works on a single line of text and the objective of this stage is to estimate a projective transformation – a 3×3 homography matrix – of the original image's region of interest to an area in which the candidate text would be rectified. The orientation detection is performed in two stages: parallel rectification and shear estimation.

Parallel rectification – A line is fitted to the centre point of every character in the text line (the centre of mass already computed before), using a least squares method, and named the *mid-line*. As used and defined here, this line will not usually correspond to any conventional typography line in the text. Possible errors or variations in the location of the characters' centre points, and so the mid-line, will not significantly affect the rectification, as it is used as an approximate guide of the direction of the text line, allowing us to define which side of the text line is the 'top' and the 'bottom' respectively.

For every character, the farthest contour points on each side of the mid-line are gathered as the top and bottom point sets respectively. On severely distorted characters, the estimated top points (and likewise the bottom points) will not exactly correspond to the actual top (and bottom) points of that character within its reference plane and orientation. It is, however, an adequate and sufficient approximation for the estimation of the top and bottom lines (see Fig. 3). Again, small variations on the location of the mid-line will not significantly affect the rectification.

A top line is then obtained by performing a least squares line fitting with RANSAC outlier removal on the computed top points. This process is repeated with the bottom points to get a bottom line. The outliers discarded during the fitting will usually correspond to the ascenders or descenders of those characters that have them (see Fig. 4).

Two additional lines are computed as follows: through every contour point of each character, a line is projected perpendicular to the mid-line. Of all these projected lines, the left-most and the right-most ones along the direction of the mid-line are kept and named the 'left' and 'right' lines. The intersection of the four computed lines (top, bottom, left and right) forms a quadrilateral with vertices A, B, C and D, labelled clockwise starting with the intersection of the left and top lines, as in the example shown in Fig. 4.

A straightforward homography $\mathbf{H}_\text{p}$ from four pairs of matching points [12] is computed so that the quadrilateral (ABCD; Fig. 4) is mapped to a rectangle (A′B′C′D′; Fig. 5). The aspect ratio and size of the target rectangle are still unknown, but not significant as the OCR engine is scale

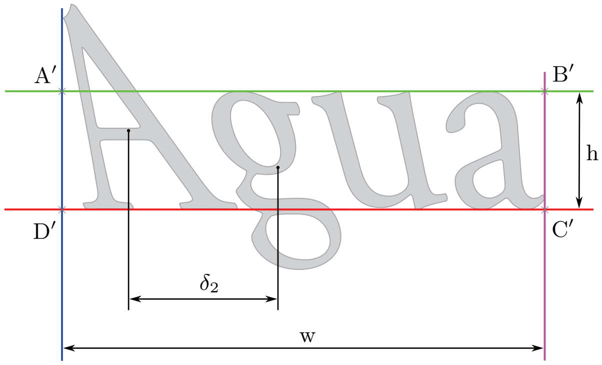


Figure 5: Image partially rectified according to the top, bottom, left and right lines. The displacement (δ_2) for the second character is also shown.

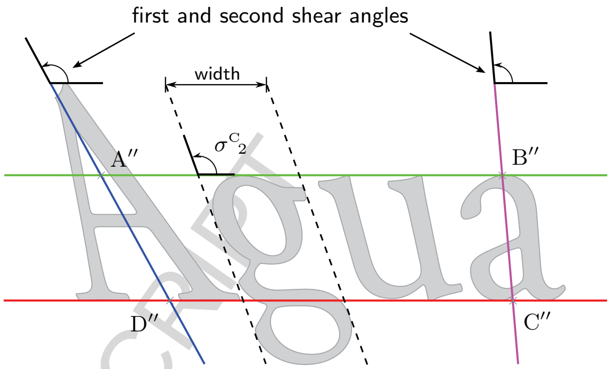


Figure 6: Quadrilateral formed after computing the two shear angles. Additionally, the *upright shear angle* (σ^{C}) for the second character is shown.

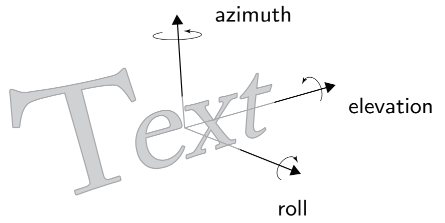


Figure 7: Axes for the rotations applied to text in our experiments.

independent. The rectified image, however, needs to have enough resolution for the OCR to operate. Hence, we define the dimensions of the target rectangle (w, h) as:

$$\mathrm{w} = \max(d(\mathrm{A}, \mathrm{B}), d(\mathrm{C}, \mathrm{D}))\ ,\ \mathrm{h} = \max(d(\mathrm{A}, \mathrm{D}), d(\mathrm{B}, \mathrm{C}))\ , \quad (5)$$

where $d(a, b)$ is the Euclidean distance between two points.

This partial rectification will transform the top and bottom lines into being horizontal and parallel, removing the distortion produced by the horizontal vanishing point. We refer to the result at this stage as the *parallel image*.

Shear estimation – A shear effect still remains in the projected text line in the parallel image due to the vertical vanishing point (this is clearly discernable in Fig. 5). As previously stated, correcting the shear has always been a challenging problem. We look at the shear angle variation of the characters within the line to perform a linear regression of angle values and obtain an accurate estimation of two shear angles at the edges of the text line, which will in turn implicitly define the vertical vanishing point.

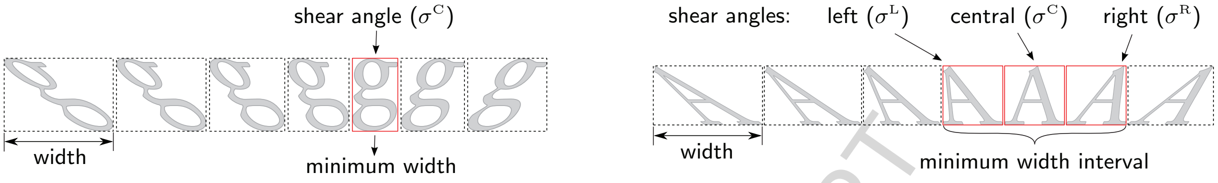
First, the characters' centre points are ordered along the x-axis in the parallel image. The horizontal distance of each character's centre point to the left-most one is called displacement δ. For the sake of clarity, the character indexes used in this section and the referenced figures will reflect this ordering. Consequently, the first character is the left-most one and its displacement is zero ($\delta_1 = 0$). Fig. 5 shows the displacement for the second character, i.e. δ_2.

Next, an upright shear angle is computed for each character which is the shear value at which the width of the character's vertical projection is minimized. Most characters have a single angle which minimizes this projection, and we refer to this as σ^{C} (see Fig. 8a), however, some characters have a range of angles, e.g. those with a triangular shape such as letters '`A`' or '`V`' (see Fig. 8b). In those cases, three candidate angles are considered: the left (σ^{L}), right (σ^{R}) and central (σ^{C}) angles of the interval, with $\sigma^{\mathrm{C}} = (\sigma^{\mathrm{L}} + \sigma^{\mathrm{R}})/2$. Thus, after any character's shear estimation, the character has either one (σ^{C}) or three (σ^{L}, σ^{C}, σ^{R}) angle estimates. It is of note that for some symbols (e.g. the forward slash – '`/`') the width minimization produces an incorrect upright shear angle estimate.

A set of 2D points comprising pairs of displacement and shear angle is constructed: $(\delta_i, {\sigma^{\mathrm{C}}}_i)$, $(\delta_i, {\sigma^{\mathrm{L}}}_i)$ and $(\delta_i, {\sigma^{\mathrm{R}}}_i)$, $i = 1, \ldots N$. Again, linear regression is performed on these points, including RANSAC-based outlier removal which will discard those shear estimations that do not fit with the shear angle variation within the text line. For example, in Fig. 9 the first letter '`A`' has three angle estimates and two of them are discarded as outliers, while the rest of the letters only have one angle estimation. The fitted line is then used to calculate two shear angles at the ends of the text line (i.e. at δ_1 and δ_N, as also illustrated in Fig. 9).

On an implementational note, the upright shear angle can be efficiently computed using a variation of the Rotating Calipers paradigm [26]. In its standard form, it is used to compute the diameter of a convex polygon by minimizing the distance between two parallel lines that are rotated around antipodal vertex pairs. Consequently, we operate on the character's convex hull, but we select the pair of lines with minimum horizontal distance (i.e. distance along the x-axis direction). The angle of these lines with respect to the x-axis is the character's upright shear angle. In Fig. 6, the upright shear angle for the second character (i.e. ${\sigma^{\mathrm{C}}}_2$) is shown along with the parallel lines used to minimize the width. The estimated first and second shear angles of the text line are also portrayed.

Once both shear angles are obtained, two lines can be


shear angle (σ^{C})
width
minimum width
shear angles:
left (σ^{L})
central (σ^{C})
right (σ^{R})
width
minimum width interval

(a) one shear angle estimation

(b) three shear angles estimation

Figure 8: Shear estimation for one character.

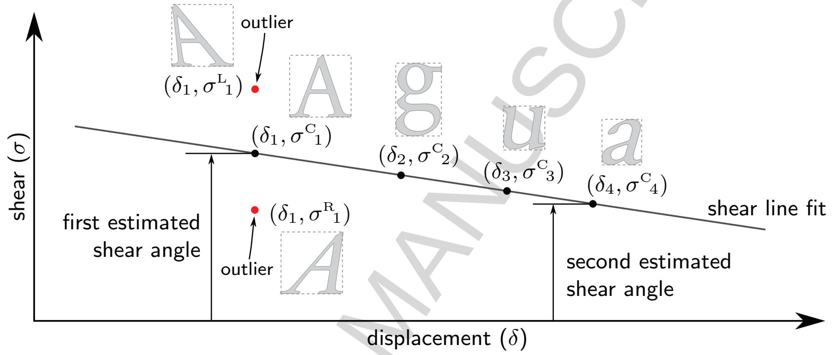

outlier
$(\delta_1, \sigma^{\mathrm{L}}{}_1)$
$(\delta_1, \sigma^{\mathrm{C}}{}_1)$
$(\delta_2, \sigma^{\mathrm{C}}{}_2)$
$(\delta_3, \sigma^{\mathrm{C}}{}_3)$
$(\delta_4, \sigma^{\mathrm{C}}{}_4)$
shear line fit
shear (σ)
first estimated shear angle
$(\delta_1, \sigma^{\mathrm{R}}{}_1)$
outlier
second estimated shear angle
displacement (δ)

Figure 9: Shear angles estimation – from the alternative shear candidates of the first letter ($\sigma^{\mathrm{L}}{}_0$, $\sigma^{\mathrm{C}}{}_0$ and $\sigma^{\mathrm{R}}{}_0$) the wrong ones (in red) are discarded as outliers by the RANSAC line fitting.

defined on both sides of the text line. They pass through the centre of the left-most and right-most characters respectively and forming an angle with respect to the x-axis equal to the computed shear angles. These lines intersect the rectified top and bottom lines defining a quadrilateral (A″B″C″D″, as shown in the example in Fig. 6). A homography $\mathbf{H}_{\mathrm{s}}$ mapping this new quadrilateral to a rectangle is computed. The result of this transformation is the rectified image.

Thus, the full rectifying homography for the original image is the combination of both partial rectifications:

$$\mathbf{H} = \mathbf{H}_{\mathrm{s}}\, \mathbf{H}_{\mathrm{p}} \tag{6}$$

4. Experiments and Results

In our first set of experiments, we used synthetic images to systematically evaluate the performance of our perspective rectification method along all possible viewpoint orientations. In the second set, examples of natural scene images were used to illustrate and evaluate the proposed method further. Throughout the experiments, we compare off-the-shelf OCR recognition accuracy on the unrectified images, on images post-rectification by our proposed method, and on images post-rectification by the method of Myers et al. [8]. The nomenclature we use for the axes is illustrated in Fig. 7: roll for in-plane rotation, elevation for the axis aligned with the text line direction and azimuth for the vertical axis with respect to the text.

It should be noted that we did not use the ICDAR 2003 Robust Reading dataset [27] or the Street View Text dataset [28], as neither contains text captured at perspective views, hence they are ill-suited to our purpose here.

4.1. Comparative evaluation on synthetic data

Our synthetic images simulate text appearing at different orientations. As text segmentation is error-free on the synthetic images, the result will not be affected by possible text localisation mistakes that would arise from using real-world images, and so we obtain an accurate performance figure of the proposed perspective recovery method alone.

To provide a realistic sample of texts among those usually encountered in a typical city environment, we use all the words (with 3 or more characters) from the groundtruth dataset of the ICDAR 2011 Robust Reading Competition (challenges 1 and 2) [6, 7], giving us a set of 3225 short phrases and single words. These are rotated along all possible orientations in the range [−90°, 90°] in 5° increments in each of the three axes, resulting in a total of over 162 million images; thus each image contains one phrase in a particular orientation. A selection of the images generated is shown in Fig. 12.

Every image is then rectified with our proposed perspective recovery method to obtain a fronto-parallel image. For comparison purposes, Myers et al.'s method [8] is also implemented and used to recover the image. An additional groundtruth baseline image is obtained by rectifying the original image with the known groundtruth orientation data. Then, the original image, the recovered images from each method respectively and the groundtruth baseline image are run through an OCR engine[1]. For each recognized text an accuracy measure is obtained, based on the Levenshtein distance, which represents the difference between the groundtruth and the recognized text normalized by the length of the groundtruth text, i.e.

$$\text{accuracy}(\mathrm{R}, \mathrm{G}) = 1 - \frac{\min\left(\mathbb{L}(\mathrm{R}, \mathrm{G}), \#\mathrm{G}\right)}{\#\mathrm{G}} \,, \qquad (7)$$

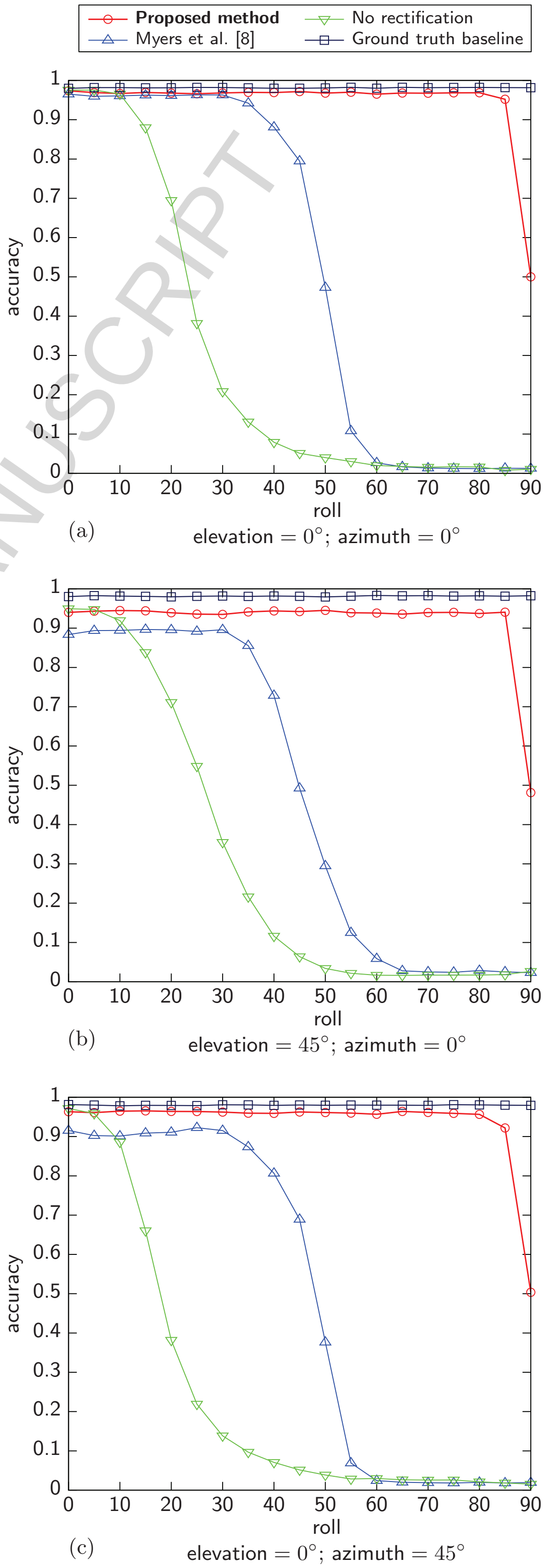
where R is the text recognized by the method under examination, G is the groundtruth text, $\mathbb{L}(x, y)$ is the Levenshtein distance between two texts, and $\#x$ is the length of a text string. With this measure, 0 is a complete miss and 1 is a perfect recognition.

For each possible orientation, the average accuracy over all the phrases is computed which gives a rectification performance evaluation from the recognition point of view. The groundtruth baseline helps get an indication of the recognition accuracy and optical resolution limit of the OCR engine. Even with a perfect rectification, some non-dictionary words are never recognized properly and, in extreme orientations, some resulting images might not have enough resolution for the OCR to operate (see e.g. Figs. 12i, 12m or 12p, where the side of the text is blurred).

In Fig. 10, where the effect of roll is studied, Fig. 10a shows the performance of the recovery when only in-plane rotations are considered, while Figs. 10b and 10c evaluate the combination of roll with elevation and azimuth at 45° respectively. As shown in the results, our method is not affected by text's in-plane rotation, yielding a constant recognition accuracy for the whole range of roll angles except when roll = 90°. The case of roll = 90° is particular because the mid-line is vertical (or close to) and the 'up' direction is not clear. Although the perspective distortion is properly corrected, the text might be rectified upside down (see e.g. Figs. 12l or 14h), which produces an incorrect recognition. Upside down text could be easily detected by performing two OCR recognitions: on the rectified image rotated at 0° and at 180°, and keeping the one with higher OCR confidence. As the focus of this work is on the perspective rectification technique, we present the method as is, without this post-processing correction step for this specific and extreme case.

The results in Fig. 10 are consistent in our experiments for the full range of elevation and azimuth values. Consequently, for ease of exposition and presentation, we will focus on demonstrating the effect of azimuth and elevation changes only, and the following graphs will all have roll fixed at 0°.

Figure 11 studies the effect of azimuth and elevation against each other. The left column portrays the variation of azimuth for fixed values of elevation (0°, 30° and 45°



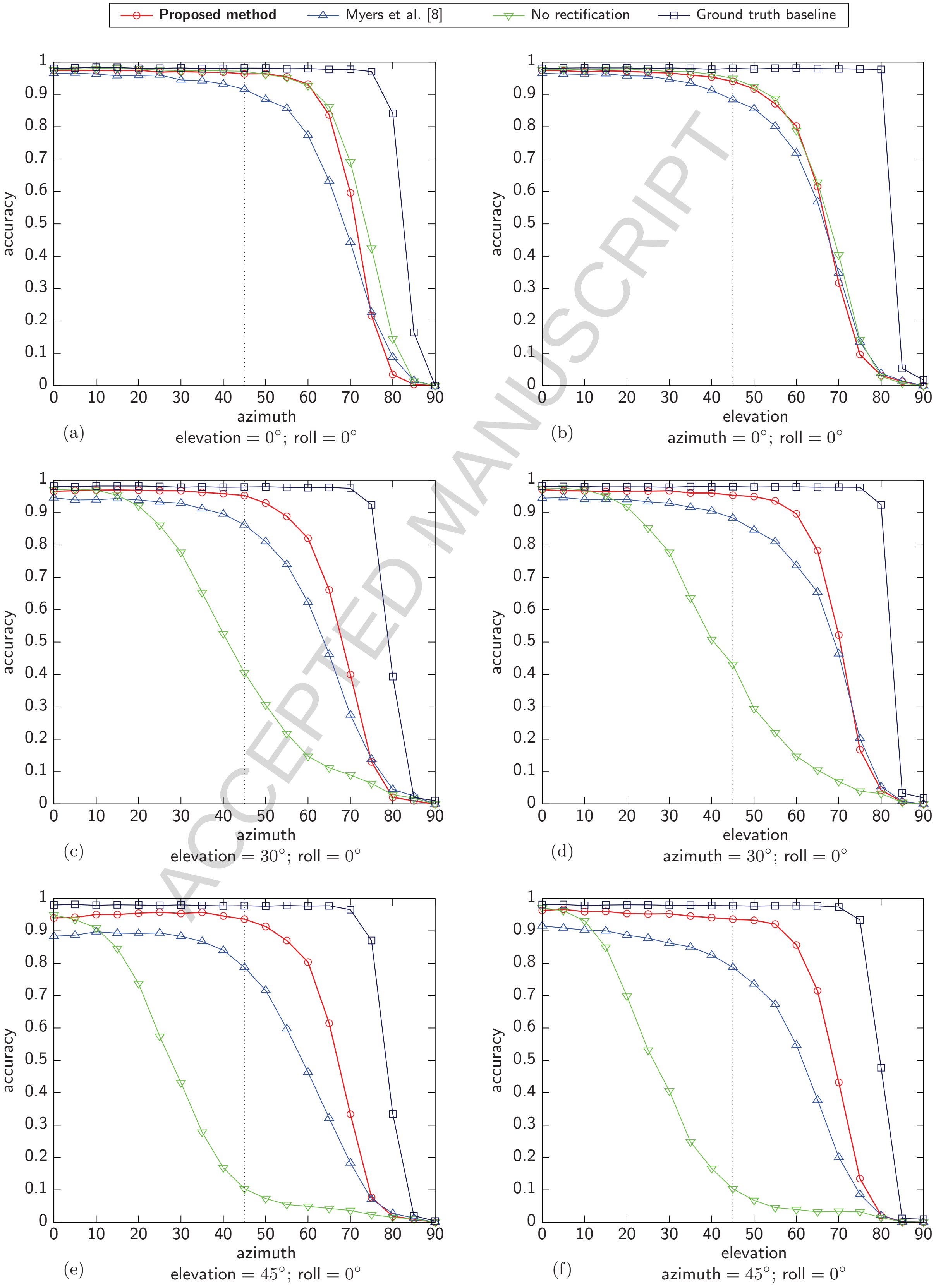
(a) elevation = 0°; azimuth = 0°

(b) elevation = 45°; azimuth = 0°

(c) elevation = 0°; azimuth = 45°

Figure 10: The effect of roll on recognition accuracy.

[1] We used Tesseract OCR: http://code.google.com/p/tesseract-ocr/



(a) elevation = 0°; roll = 0°

(b) azimuth = 0°; roll = 0°

(c) elevation = 30°; roll = 0°

(d) azimuth = 30°; roll = 0°

(e) elevation = 45°; roll = 0°

(f) azimuth = 45°; roll = 0°

Figure 11: The effect of azimuth and elevation on recognition accuracy.

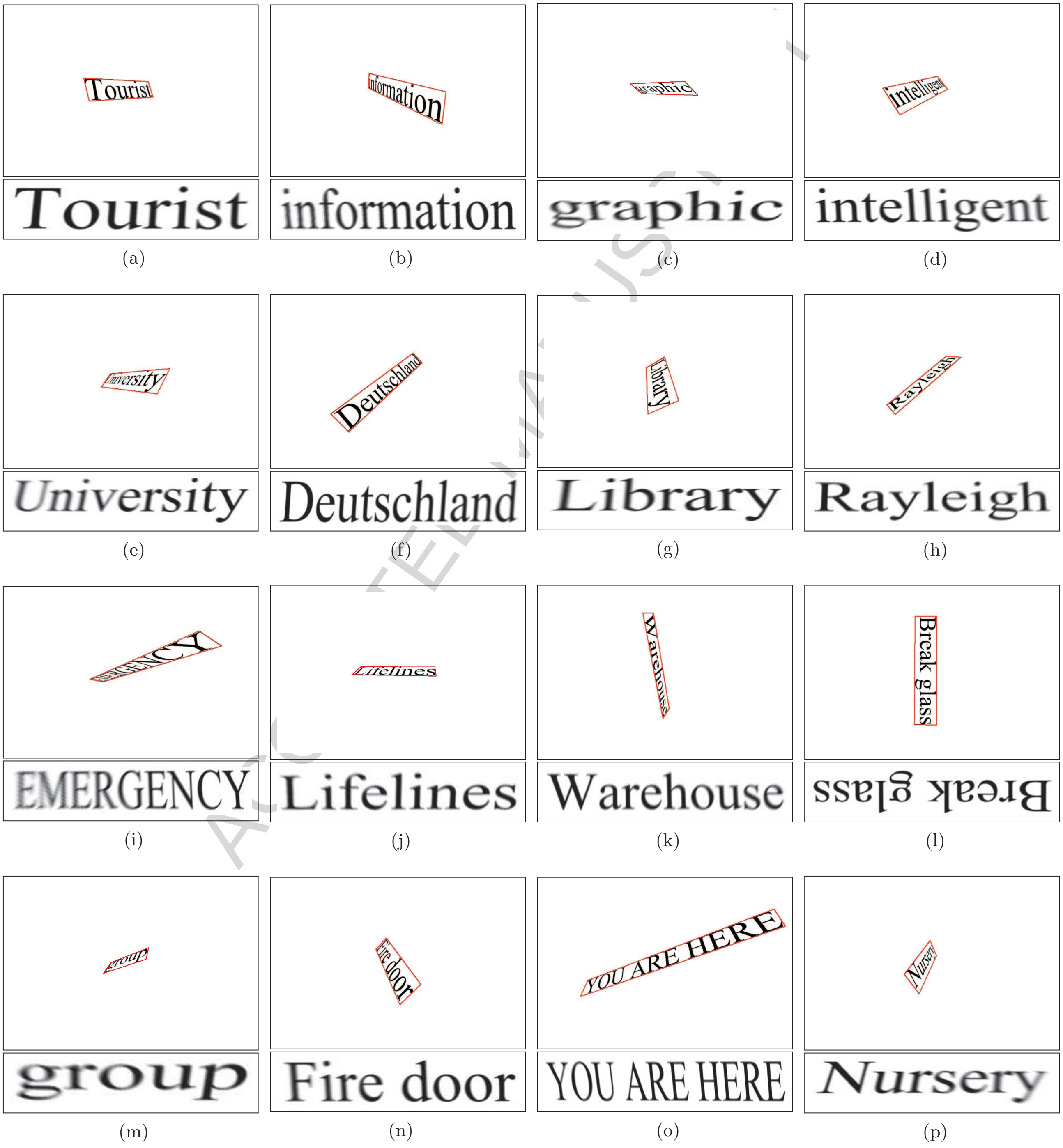


Figure 12: A selection of the synthetic images used in the experiments, along with their *estimated* orientation (red box) and corresponding rectified image.

– Figs. 11a, 11c and 11e respectively) and likewise, the right column displays the variation of elevation for fixed values of azimuth (0°, 30° and 45° – Figs. 11b, 11d and 11f respectively). Considering each axis separately, any angle of roll, up to 50° in azimuth and up to 45° in elevation yield an almost perfect average recognition accuracy of 0.96 after recovery. This recognition accuracy is maintained for any combination of angles under 45°. The method also achieves a very good recognition accuracy (above 0.8) for any combination of angles up to 60°. Compared to the results reported in Myers et al. [8], our proposed method shows an increase in recognition accuracy for a wider range of angles.

As expected, the OCR engine alone deals in a very limited way with perspective distortion. Any changes in roll, azimuth or elevation quickly introduce recognition errors after around 20–25°. In our experiments, the method by Myers et al. [8] performs well (more than 0.9 accuracy) with roll until 40°, in azimuth up to 45° and in elevation up to 30°, when each angle is studied separately. The differences in the methods are more apparent when combined rotations are introduced. For example, looking at elevation changes alone (Fig. 11b), the three methods perform similarly. However, when combined with azimuth (Figs. 11d and 11f) the proposed method retains the same accuracy (0.96 average accuracy up to 45°), while the OCR fails quickly and Myers et al.'s method accuracy degrades rapidly.

Another parameter that affects recognition accuracy after rectification is word length, measured as the number of non-whitespace characters of a given text line. The RANSAC algorithm needs a certain ratio of inlier vs. outlier points to accurately estimate the top and bottom lines. To establish the effect of word length in rectification accuracy, Fig. 13 shows the average recognition accuracy per word length, for all values of roll, azimuth and elevation under 45°. The proposed method performs best (with more than 0.98 average recognition accuracy) with words of at least 6 characters, The recognition accuracy is also very good (above 0.9) with words as short as 4 characters. As a reference, Table 1 illustrates the distribution of word lengths in the set of words used in our experiment.

length	count	length	count
3	376	10	141
4	640	11	85
5	504	12	53
6	460	13	31
7	430	14	18
8	269	15+	24
9	194	**total**	3225

Table 1: Word length distribution in the synthetic text dataset.

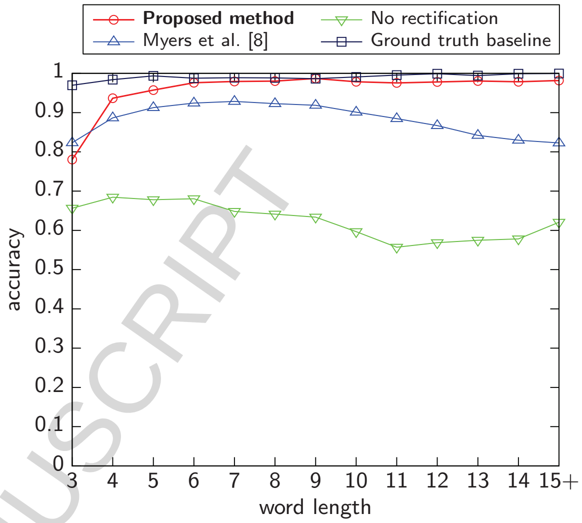


Figure 13: The effect of word length on recognition accuracy.

4.2. Natural scene images

The first experiment was designed to evaluate the accuracy of the rectification step alone, assuming a perfect text detection result. Real world images feature complex backgrounds, uneven lighting and noise, which can confuse the text segmentation stage and occasionally produce wrongly labelled text regions. To obtain a measure of the method performance for real, everyday scenarios, a set of 120 natural scene images were used to evaluate the system. They contain scene text from shop names and signs taken at various orientations, comprising several typefaces (e.g. Figure 14 shows several examples from the image set after applying our proposed method, illustrating the resulting bounding boxes obtained after the text detection stage (referred to in Section 3.1) and corresponding rectified images. The images were manually annotated to obtain a groundtruth of the text present in them. Table 2 shows a comparison of the average recognition accuracy, using (7), on the unrectified images, and after rectifying with Myers et al.'s [8] method and the proposed method, with the latter showing marked improvement.

No rectification	0.25
Myers et al. [8]	0.40
Proposed method	0.87

Table 2: Average OCR recognition accuracy on the real-world image set.

Given the unconstrained way in which our method extracts the top and bottom lines, it is specially well suited to correct any kind of text's in-plane rotation, as seen in the results. Furthermore, our shear angles computation (taking into account the variation of shear across the whole line) allows us to correctly detect the orientation of words that end in non-square letters (e.g. see the 'Y' in Figs. 12g, 12i, 12o, 14a, 14n and 14o, the 'T' in Figs. 12a, 14g and 14o, or the 'W' in Figs. 12k, 14e and 14o). In these cases, a naïve box fitting approach would fail. Text lying on the ground, or far above the camera introduce big shear distortions which are also properly corrected with this technique (as seen in Figs. 14b, 14f and 14m).

4.3. Speed

In our implementation, text extraction, including segmentation, grouping and perspective estimation, performed on an Intel Core i7-2600 processor, achieved real-time performance of 20 fps on 1280×720 video sequences. The orientation detection stage (as explained in Section 3.3) requires, on average, 0.1 ms per text line. As a reference, our implementation of Myers et al.'s [8] method needs 20 ms per text line.

5. Conclusion and future work

We presented here a technique for perspective recovery of text in natural scenes. Aimed at scene text, it focuses on isolated words or short sentences, as found on billboards, posters, shop names, street signs etc. It is a geometrical approach that relies exclusively on the contours of segmented characters and thus does not depend on higher level structures in the text such as borders or paragraphs. It is also fast, allowing for a real-time implementation. Experiments and comparative results show an increased accuracy in text recognition after recovery, compared to the current state-of-the-art 3D text recovery technique.

The proposed method outperforms previous approaches in scene text perspective recovery, however, its current limitations are mainly related to the quality of the input into it, i.e. the earlier stage of text segmentation. Noisy regions in the image, which would be incorrectly labelled as text can confuse the top/bottom lines estimation and upright shear angle computation. In extreme orientations, the available resolution of text in the image is limited. Low resolution can cause the RANSAC line fitting method to pick up the wrong combination of points for the top and bottom line estimation. This happens on words with false slopes, i.e. words or phrases with uneven distributions of tall and short characters. For example, we have found that the phrase '`lifelines`' is specially challenging for our method, as the tall letters are all distributed at the beginning of the word. On some orientations, the estimated top line can lie slanted between the tops of the first '`l`' and the last '`s`' respectively, rendering the perspective estimation incorrect. Yet, in many cases it is still recognisable by the OCR.

Our future plan is to address the current shortcomings of our method. We will look into improving our text grouping algorithm, aiming to achieve a better clustering of candidate text regions into text lines. If the line formation was also to provide clues about higher level structures, such as paragraphs, that information could also be used to improve the understanding of the scene as a whole.

References

[1] C. Merino-Gracia, K. Lenc, M. Mirmehdi, A Head-Mounted Device for Recognizing Text in Natural Scenes, in: CBDAR'11, vol. 7139 of *LNCS*, Springer Berlin / Heidelberg, 29–41, 2012.

[2] J. Gao, J. Yang, An adaptive algorithm for text detection from natural scenes, in: CVPR'01, vol. 2, II:84–89, 2001.

[3] C. Mancas-Thillou, S. Ferreira, J. Demeyer, C. Minetti, B. Gosselin, A multifunctional reading assistant for the visually impaired, JIVP (2007) 5:1–11.

[4] I. Posner, P. Corke, P. Newman, Using text-spotting to query the world, in: IROS'10, 3181–3186, 2010.

[5] L. Li, C. L. Tan, Recognizing Planar Symbols with Severe Perspective Deformation, PAMI 32 (4) (2010) 755–762.

[6] D. Karatzas, S. Robles Mestre, J. Mas, F. Nourbakhsh, P. Pratim Roy, ICDAR 2011 Robust Reading Competition - Challenge 1: Reading Text in Born-Digital Images (Web and Email), in: ICDAR'11, 1485–1490, 2011.

[7] A. Shahab, F. Shafait, A. Dengel, ICDAR 2011 Robust Reading Competition Challenge 2: Reading Text in Scene Images, in: ICDAR'11, 1491–1496, 2011.

[8] G. K. Myers, R. C. Bolles, Q.-T. Luong, J. A. Herson, H. B. Aradhye, Rectification and recognition of text in 3-D scenes, IJDAR 7 (2005) 147–158.

[9] Y. Y. Tang, S.-W. Lee, C. Y. Suen, Automatic document processing: A survey, PR 29 (12) (1996) 1931–1952.

[10] G. Nagy, Twenty years of document image analysis in PAMI, PAMI 22 (1) (2000) 38–62.

[11] T. Saba, G. Sulong, A. Rehman, Document image analysis: issues, comparison of methods and remaining problems, AIR 35 (2011) 101–118.

[12] R. I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, second edn., 2004.

[13] M. Pilu, Extraction of illusory linear clues in perspectively skewed documents, in: CVPR, 363–368, 2001.

[14] P. Clark, M. Mirmehdi, Recognising text in real scenes, IJDAR 4 (2002) 243–257.

[15] P. Clark, M. Mirmehdi, Rectifying perspective views of text in 3D scenes using vanishing points, PR 36 (11) (2003) 2673–2686.

[16] A. B. Cambra, A. Murillo, Towards robust and efficient text sign reading from a mobile phone, in: ICCV Wshps., 64–71, 2011.

[17] N. Stamatopoulos, B. Gatos, I. Pratikakis, S. Perantonis, Goal-Oriented Rectification of Camera-Based Document Images, IEEE TIP 20 (4) (2011) 910–920.

[18] J. Liang, D. DeMenthon, D. Doermann, Geometric Rectification of Camera-Captured Document Images, PAMI 30 (4) (2008) 591–605.

[19] X. Chen, L. Yang, J. Zhang, A. Waibel, Automatic detection and recognition of signs from natural scenes, IEEE TIP 13 (1) (2004) 87–99.

[20] T. Yamaguchi, M. Maruyama, H. Miyao, Y. Nakano, Digit recognition in a natural scene with skew and slant normalization, IJDAR 7 (2005) 168–177.

[21] C. Merino, M. Mirmehdi, A Framework Towards Realtime Detection and Tracking of Text, in: CBDAR'07, 10–17, 2007.

[22] Y.-F. Pan, X. Hou, C.-L. Liu, Text Localization in Natural Scene Images Based on Conditional Random Field, in: ICDAR'09, 6–10, 2009.

[23] B. Epshtein, E. Ofek, Y. Wexler, Detecting text in natural scenes with stroke width transform, in: CVPR, 2963–2970, 2010.

[24] A. Targhi, E. Hayman, J. Eklundh, M. Shahshahani, The Eigen-Transform & Applications, in: ACCV, I:70–79, 2006.

[25] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, Springer-Verlag, second edn., 2000.

[26] G. Toussaint, Solving Geometric Problems with the Rotating Calipers, in: IEEE MELECON'83, 10–17, 1983.

[27] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, ICDAR 2003 Robust Reading Competitions, in: ICDAR'03, 682–687, 2003.

[28] K. Wang, B. Babenko, S. Belongie, End-to-end scene text recognition, in: ICCV'11, Barcelona, Spain, 1457–1464, 2011.

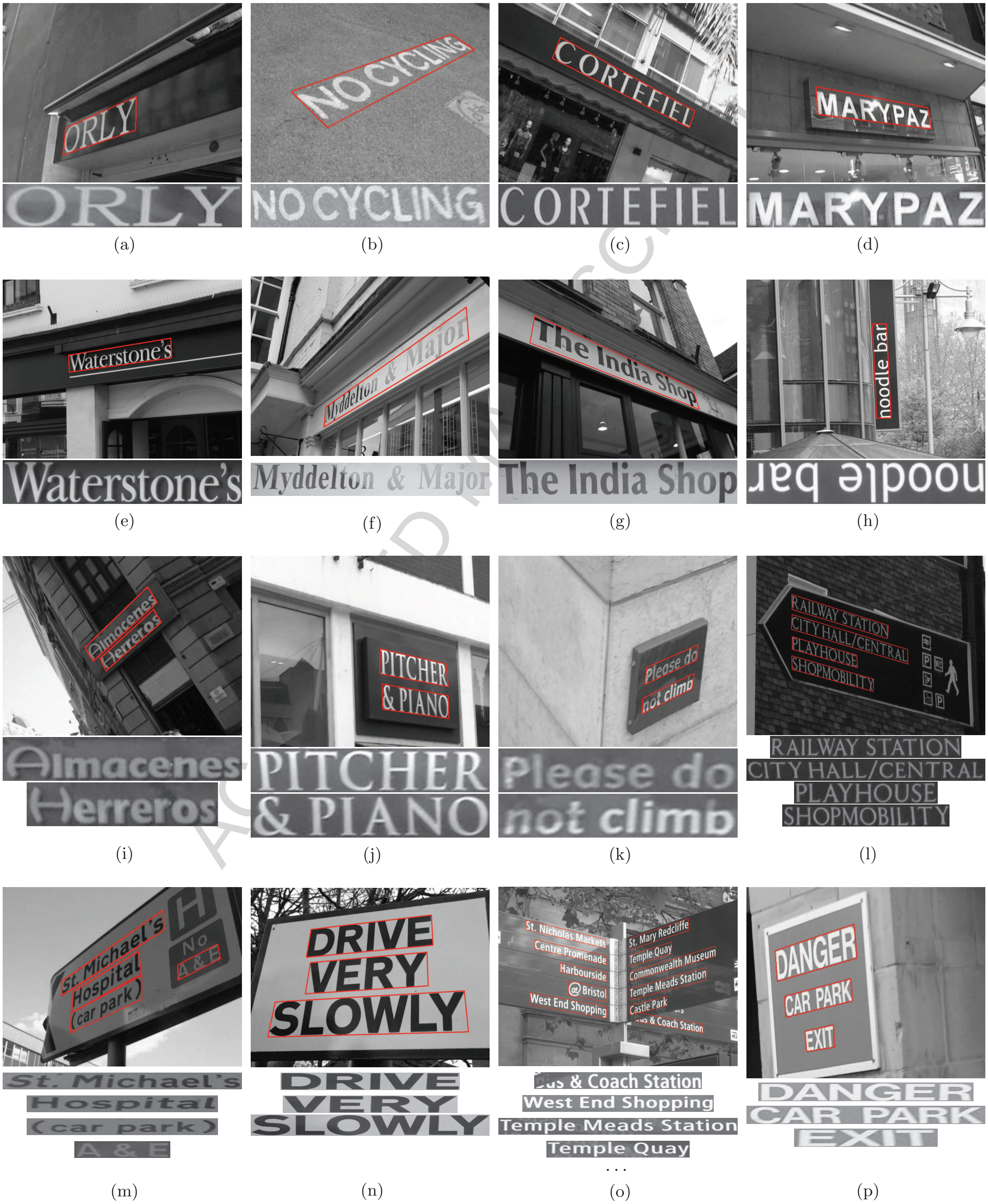
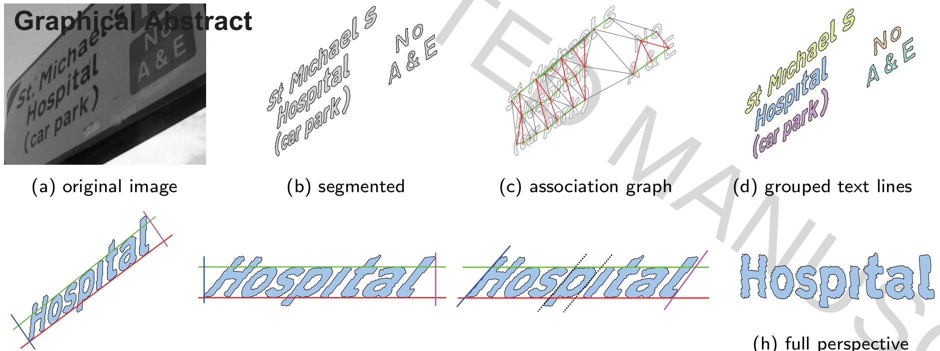


Figure 14: A selection of real world images with scene text, along with the text's estimated orientation (red box) and rectified image.

Graphical Abstract



(a) original image

(b) segmented

(c) association graph

(d) grouped text lines

(e) parallel estimation

(f) parallel image

(g) shear estimation

(h) full perspective rectification

Highlights

- We present a method for perspective recovery of text in natural scenes.
- It relies on the characters' geometry to estimate a rectifying homography.
- The proposed method is efficient and fast.
- Comparative results show improved recognition accuracy against the state-of-the-art.