Universidad de La Laguna

Escuela Superior de Ingeniería y Tecnología
Sección de Ingeniería Informática
Departamento de Ingeniería Informática y de Sistemas

Tesis doctoral

Desarrollo de un sistema de lectura de textos sobre imágenes de video

Development of a text reading system on video images

Carlos Merino Gracia

Mayo de 2015

D. JOSÉ FRANCISCO SIGUT SAAVEDRA, Doctor por la Universidad de La Laguna y Profesor Titular de Universidad del Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna.

y

D. JOSÉ LUIS GONZÁLEZ MORA, Doctor por la Universidad de La Laguna y Profesor Titular de Universidad del Departamento de Fisiología de la Universidad de La Laguna.

CERTIFICAN:

que D. Carlos Merino Gracia, Ingeniero en Informática, ha realizado bajo nuestra dirección la presente Tesis, titulada "Desarrollo de un sistema de lectura de textos sobre imágenes de video", para optar al grado de Doctor por la Universidad de La Laguna.

Con esta fecha, autorizamos la presentación de la misma.

En La Laguna, a 20 de Mayo de 2015.

El Director, El Codirector,

José Francisco Sigut Saavedra José Luis González Mora

A Nadia,
tu eres mi inspiración,
tu lo has hecho posible.

A Nieves y Juan.
A Javier y Miguel.

Acknowledgements

I would like to start thanking Dr José Luis González Mora with whom this PhD journey began. His vision and ideas have been an inspiration, and the funding he obtained made this project possible. Thank you to Dr José Francisco Sigut Saavedra for agreeing to supervise me, for his patience, encouragement, support and helpful discussions during the course of the years. Last but not least, my special thanks go to Prof Majid Mirmehdi without whom this PhD would not have been possible and who helped me steering this work towards producing academic results. Furthermore, he made me feel at home in Bristol and also opened up an unexpected career path for which I am very grateful and indebted.

I am particularly grateful to Oscar Casanova González. His engineering ability and technical insights have been an invaluable resource since the early days of this project. And in the end, when all hope had faded, he gave me the key to finishing this endeavour. I wish to express my appreciation towards Antonio Rodríguez Hernández for being so generous and altruistic. Thank you to Miguel Torres Gil for, among other things, introducing me into GPGPU programming. And I would like to acknowledge the work by Aneliya Stoyanova on the industrial design of the reading hat.

In Bristol, I also met many amazing researchers who helped me develop professionally and personally. The numerous fruitful discussions – technical or otherwise – with Ronghua Yang, Alexander Davies, Jack Greenhalgh, Chris Beck, Karel Lenc, Daniel Blueman, Sion Hannuna, Austin Gregg-Smith, Teesid Leelasawassuk and Osian Haines have enriched my own research considerably. Thank you all.

I wish to acknowledge the advice given by Francisco de Sande during the latter stages of this adventure. I greatly appreciated the administrative assistance provided by Amalia Báez García. I would also like to thank Dan Fairs, Andy Littledale, Lee Carre and Ted Littledale for not forgetting about my PhD during the turmoil of a company acquisition.

Thank you to my parents, Nieves Gracia Lezcano and Juan Julian Merino Rubio, and my brothers, Javier Merino Gracia and Miguel Merino Gracia, for their support and encouragement. And finally, I would like to thank my wife and best friend, Nadia Khelaifat, for believing in me and making everything possible. She was there for me during the highs and the lows, she was always happy to discuss any PhD detail or carefully analyse and proof read what I produced; she did not let me lose my motivation – or my mind! Thank you for pushing me, for getting me out of the many dead ends and for not letting me give up.

Contents

List of Figures

List of Tables

Acronyms

AT	Adaptive Thresholding	13
CCD	Charge-Coupled Device	3
CC	Connected Component	13
CNN	Convolutional Neural Network	5
CRF	Conditional Random Field	5
DIA	Document Image Analysis	3
IBM	International Business Machines Corporation	3
ICDAR	International Conference on Document Analysis and Recognition	4
IMR	Intelligent Machines Research Corporation	2
MSER	Maximally Stable Extremal Region [41]	5
NCC	Normalised Cross Correlation	8
OCR	Optical Character Recognition	2
PF	Particle Filter	7
RANSAC	Random Sample Consensus [19]	21
RCA	Radio Corporation of America	2
SFT	Stroke Feature Transform	5
SIFT	Scale Invariant Feature Transform [36]	8
SWT	Stroke-Width Transform	4
TBB	Intel's Threading Building Blocks	31
TTS	Text-To-Speech	3
UKF	Unscented Kalman Filter [80]	10

Abstract

Since the early days of computer science researchers sought to devise a machine which could automatically read text to help people with visual impairments. The problem of extracting and recognising text on document images has been largely resolved, but reading text from images of natural scenes remains a challenge. Scene text can present uneven lighting, complex backgrounds or perspective and lens distortion; it usually appears as short sentences or isolated words and shows a very diverse set of typefaces. However, video sequences of natural scenes provide a temporal redundancy that can be exploited to compensate for some of these deficiencies. Here we present a complete end-to-end, real-time scene text reading system on video images based on perspective aware text tracking.

The main contribution of this work is a system that automatically detects, recognises and tracks text in videos of natural scenes in real-time. The focus of our method is on large text found in outdoor environments, such as shop signs, street names and billboards. We introduce novel efficient techniques for text detection, text aggregation and text perspective estimation. Furthermore, we propose using a set of Unscented Kalman Filters (UKF) to maintain each text region's identity and to continuously track the homography transformation of the text into a fronto-parallel view, thereby being resilient to erratic camera motion and wide baseline changes in orientation. The orientation of each text line is estimated using a method that relies on the geometry of the characters themselves to estimate a rectifying homography. This is done irrespective of the view of the text over a large range of orientations. We also demonstrate a wearable head-mounted device for text reading that encases a camera for image acquisition and a pair of headphones for synthesized speech output.

Our system is designed for continuous and unsupervised operation over long periods of time. It is completely automatic and features quick failure recovery and interactive text reading. It is also highly parallelised in order to maximize the usage of available processing power and to achieve real-time operation. We show comparative results that improve the current state-of-the-art when correcting perspective deformation of scene text. The end-to-end system performance is demonstrated on sequences recorded in outdoor scenarios. Finally, we also release a dataset of text tracking videos along with the annotated ground-truth of text regions.

Chapter 1

Introduction

The Virtual Acoustic Space Group, part of the Neurochemistry and Neuroimaging Laboratory,[1] is a multidisciplinary Research and Development Team. Under the direction of Dr José Luis González-Mora it has specialized in sensory substitution and, in particular, the perception of the environment using sounds. Its charter is to develop devices which improve the quality of life of people who are blind or visually impaired, enabling them to perceive and interact with their environment in a richer capacity than what their disability allows them. In this context, it was envisaged that a text reading assistant for people who are blind could be developed.

This work focuses on the computer vision aspects of the problem. It is an exploration of the techniques needed to understand text in videos of natural scenes. At the University of La Laguna, this PhD has been supervised by Dr José Francisco Sigut Saavedra from the Department of Computer and Systems Engineering.[2] Additionally, an important part the research was carried out during my stay at the University of Bristol in collaboration with the the Visual Information Laboratory[3] and under the supervision of Prof Majid Mirmehdi.

This PhD is submitted by publication ("compendio de publicaciones"). Therefore, to comply with the University's regulations, the dissertation has the following structure: this chapter will introduce the problem and establish the context by giving a brief – and by no means exhaustive – background review, as this would be beyond the scope of the present work. Next, I will describe the PhD's scope and research objectives by outlining the work carried out for each one of the articles published during the course of this project. These works have been edited into a coherent narrative that covers their methodology and results in Chapters 2 and 3, respectively. Chapter 4 will present the conclusions, open research lines and future work. Finally, as a reference, Appendices A to E contain the full text copies of the articles published as a result of this PhD.

1.1. The problem of making a computer read

Accessing textual information in our environment is crucial for our daily lives. There is an obvious need for technology that can automatically extract and process this information for the benefit of humans that might have difficulties accessing it, for instance, assistance for blind people, tourist aiding devices, or road sign detection. In fact, the first devices developed for automatic text reading were aimed to assist blind people. Reading has become a second nature for many of us, and the area of document analysis and recognition has been an important focus area in computer

[1] https://nf.ull.es

[2] http://www.departamentos.ull.es/view/departamentos/inginformatica/Inicio

[3] http://www.bris.ac.uk/vi-lab/

vision and machine learning since the early days of computer science. What is more, the first automatic text recognition attempts predate programmable computers.

But if we want to build a device to *read*, we first have to define what we consider *reading*. For example, when looking at our surrounding, we could argue that reading involves understanding where the text is and its context, in order to be able to communicate its contents. However, it can also simply be defined as the translation of printed characters into a different representation. How much understanding is needed and how important is the communication aspect? It is a crucial question in order to develop an assistant device for blind people, as we want to provide an useful interface with the real world. Ideally, we should aim towards a device that would behave as closely as possible to a human assistant. This section will explore how far we have come towards a true reading assistant, and how the definition of reading machine has evolved over the years up to the current state-of-the-art.

1.1.1. The early reading machines

In 1914, Fournier d'Albe invented a reading device aimed at assisting people with visual impairments. It was called the *optophone* [20] and can be considered the first reading machine in history. The device converted characters to audio signals using a technique named sonification [32]. It produced a characteristic, non-speech sound for each printed symbol that the users learnt to associate back to the characters, and thus, with adequate training, they were able to read books, newspapers, etc. This device did not recognise any letters as such, but simply mapped certain characteristics of the letters to sound patterns. Thus, it was the user's brain who carried out the actual recognition. In 1929, Gustav Tauschek invented the first Optical Character Recognition (OCR) device. It was an electro-mechanical device that employed a drum engraved with patterns of letters and a photoelectric cell that acted as a primitive matching mechanism between the patterns and the images. By the late 1940s, the Radio Corporation of America (RCA) had developed two different prototypes of reading machines for the assistance people with visual impairment. One of them employed a pencil with an optical scanner that would match characters one by one and translate them into the sound of each individual letter.

Some observations are apparent from these early mechanical or electro-mechanical devices. The main application areas pursued by the inventors were *(i)* information extraction for automatic transmission (i.e., telegraph) and *(ii)* blind people assistance. Secondly, these prototypes and the interest in this kind of technology pre-date digital programmable computers. No commercially viable product was made, but some of the initial techniques were explored, such as template matching, which limited the recognition ability to one font in one size. And finally, the devices required the user to perform numerous manual operation of the device: aligning the sensor with the text or following the printed characters. In this case, "reading" was just the process of translating individual characters to a different representation (sound).

1.1.2. The development of Optical Character Recognition (OCR)

With the advent of the digital programmable computer in the 1950s, and over the following three decades, the OCR technology took off with commercially viable products. Intelligent Machines Research Corporation made the first commercial OCR device in 1959 after almost a decade

of research [69, p. 12]. During this period, OCR evolved from single font specialized devices that processed text printed on tape or cards to multi-font document processing systems. IBM produced a few notable devices during this time (e.g., the IBM 141B, IBM 1975 or the IBM 1287), along with other corporations such as CDC or NCR. OCR also blossomed as a research area and it required advances in a broad range of disciplines, from acquisition technology to image processing. A detailed review of the OCR technology is out of the scope of this introduction, but the interested reader can refer to the work by Mori et al. [52] for a description of the techniques developed during this period.

The development of OCR and, simultaneously, advances in speech synthesis [70] allowed the conception of more ambitious reading machines. In 1975, Ray Kurzweil developed a device that used a flat CCD scanner, a computer unit with an omnifont OCR software and a Text-To-Speech (TTS) synthesis system. It was named the *Kurzweil's Reading Machine* [33], and it was the size of a washing machine. Speech synthesis allowed the translation of whole words and sentences from the scanned document. It was aimed at documents (e.g., books, magazines, newspapers) and still required a great deal of operation from the user, for instance putting the documents on the flat-bed scanner. However, reading had evolved from converting one symbol at a time into encoded sounds to being able to produce a spoken translation. Several desktop reading machines with a similar design are still widely available (such as the POET reader from the Spanish ONCE,[1] or the Exalibur reader from the Australian ROBOTRON[2]).

Nowadays, Optical Character Recognition (OCR) is a mature technology and is widely considered a solved problem. It is one of the most successful applications of Computer Vision and Pattern Recognition and word recognition accuracies of +99% are expected from any modern OCR engine operating on clean document images [35]. The discipline has evolved into what is called Document Image Analysis (DIA) as the problem focus evolved from letters and words recognition to the processing of whole documents, including layout and formatting analysis or extraction of non-textual content. Some selected reviews of the recent evolution of OCR can be found in [57, 68, 76].

1.1.3. Camera based scene text detection and recognition

In the late 20th century, as digital cameras started to become inexpensive, a new focus area for document image analysis started to take shape: camera based document analysis. Initially, some research effort looked into applying and adapting classical document image analysis techniques to pictured documents as a way to replace digital scanners as the acquisition mechanism [49, 62, 82]. Yet, a logical step is not only to capture document images but to use cameras to try to understand text in natural scenes or *scene text* [3, 10]. Natural scene images pose significant challenges to text understanding, such as blurred or out of focus frames, uneven lighting, complex backgrounds or perspective and lens distortion. Additionally, the text appearing in these images is usually composed of short sentences or isolated words and showing a very diverse set of typefaces. Figure 1.1 illustrates the difference between a document image and a scene text image. On the other hand, when operating on video, a continuous stream of frames provides a temporal

[1] `http://www.once.es`

[2] `http://www.sensorytools.com/`

No. 4356 April 25, 1953 NATURE 737

equipment, and to Dr. G. E. R. Deacon and the captain and officers of R.R.S. *Discovery II* for their part in making the observations.

1 Young, F. B., Gerrard, H., and Jevons, W., *Phil. Mag.*, **40**, 149 (1929).
2 Longuet-Higgins, M. S., *Mon. Not. Roy. Astro. Soc., Geophys. Supp.*, **5**, 285 (1949).
3 Von Arx, W. S., Woods Hole Papers in Phys. Oceanog. Meteor., **11** (3) (1950).
4 Ekman, V. W., *Arkiv. Mat. Astron. Fysik. (Stockholm)*, **2** (11) (1905).

MOLECULAR STRUCTURE OF NUCLEIC ACIDS

A Structure for Deoxyribose Nucleic Acid

We wish to suggest a structure for the salt of deoxyribose nucleic acid (D.N.A.). This structure has novel features which are of considerable biological interest.

A structure for nucleic acid has already been proposed by Pauling and Corey[1]. They kindly made their manuscript available to us in advance of publication. Their model consists of three interwined chains, with the phosphates near the fibre axis, and the bases on the outside. In our opinion, this structure is unsatisfactory for two reasons: (1) We believe that the material which gives the X-ray diagrams is the salt, not the free acid. Without the acidic hydrogen atoms it is not clear what forces would hold the structure together, especially as the negatively charged phosphates near the axis will repel each other. (2) Some of the van der Waals distances appear to be too small.

Another three-chain structure has also been suggested by Fraser (in the press). In his model the phosphates are on the outside and the bases on the inside, linked together by hydrogen bonds. This structure as described is rather ill-defined, and for this reason we shall not comment on it.

This figure is purely diagrammatic. The two ribbons symbolize the two phosphate–sugar chains, and the horizontal rods the pairs of bases holding the chains together. The vertical line marks the fibre axis

We wish to put forward a radically different structure for the salt of deoxyribose nucleic acid. This structure has two helical chains each coiled round the same axis (see diagram). We have made the usual chemical assumptions, namely, that each chain consists of phosphate diester groups joining β-D-deoxyribofuranose residues with 3′,5′ linkages. The two chains (but not their bases) are related by a dyad perpendicular to the fibre axis. Both chains follow right-handed helices, but owing to the dyad the sequences of the atoms in the two chains run in opposite directions. Each chain loosely resembles Furberg's[2] model No. 1; that is, the bases are on the inside of the helix and the phosphates on the outside. The configuration of the sugar and the atoms near it is close to Furberg's 'standard configuration', the sugar being roughly perpendicular to the attached base. There is a residue on each chain every 3·4 A. in the z-direction. We have assumed an angle of 36° between adjacent residues in the same chain, so that the structure repeats after 10 residues on each chain, that is, after 34 A. The distance of a phosphorus atom from the fibre axis is 10 A. As the phosphates are on the outside, cations have easy access to them.

The structure is an open one, and its water content is rather high. At lower water contents we would expect the bases to tilt so that the structure could become more compact.

The novel feature of the structure is the manner in which the two chains are held together by the purine and pyrimidine bases. The planes of the bases are perpendicular to the fibre axis. They are joined together in pairs, a single base from one chain being hydrogen-bonded to a single base from the other chain, so that the two lie side by side with identical z-co-ordinates. One of the pair must be a purine and the other a pyrimidine for bonding to occur. The hydrogen bonds are made as follows: purine position 1 to pyrimidine position 1; purine position 6 to pyrimidine position 6.

If it is assumed that the bases only occur in the structure in the most plausible tautomeric forms (that is, with the keto rather than the enol configurations) it is found that only specific pairs of bases can bond together. These pairs are: adenine (purine) with thymine (pyrimidine), and guanine (purine) with cytosine (pyrimidine).

In other words, if an adenine forms one member of a pair, on either chain, then on these assumptions the other member must be thymine; similarly for guanine and cytosine. The sequence of bases on a single chain does not appear to be restricted in any way. However, if only specific pairs of bases can be formed, it follows that if the sequence of bases on one chain is given, then the sequence on the other chain is automatically determined.

It has been found experimentally[3,4] that the ratio of the amounts of adenine to thymine, and the ratio of guanine to cytosine, are always very close to unity for deoxyribose nucleic acid.

It is probably impossible to build this structure with a ribose sugar in place of the deoxyribose, as the extra oxygen atom would make too close a van der Waals contact.

The previously published X-ray data[5,6] on deoxyribose nucleic acid are insufficient for a rigorous test of our structure. So far as we can tell, it is roughly compatible with the experimental data, but it must be regarded as unproved until it has been checked against more exact results. Some of these are given in the following communications. We were not aware of the details of the results presented there when we devised our structure, which rests mainly though not entirely on published experimental data and stereochemical arguments.

It has not escaped our notice that the specific pairing we have postulated immediately suggests a possible copying mechanism for the genetic material.

Full details of the structure, including the conditions assumed in building it, together with a set of co-ordinates for the atoms, will be published elsewhere.

We are much indebted to Dr. Jerry Donohue for constant advice and criticism, especially on interatomic distances. We have also been stimulated by a knowledge of the general nature of the unpublished experimental results and ideas of Dr. M. H. F. Wilkins, Dr. R. E. Franklin and their co-workers at

(a) (b)

Figure 1.1. A comparison between a document image (a) and two natural scene images with text (b).

redundancy that can help address some of these drawbacks. For example, a blurred image can be difficult or impossible to process on its own, but as part of a sequence some frames can be ignored as there are chances that other frames in the sequence are clear.

The end-to-end problem of reading text in natural scenes has been approached by focusing on the different stages of the pipeline at first: text detection, or the process of telling which areas of the image contain text; text extraction, or selecting and post-processing those pixels of the image that form part of the text; and finally text recognition, where the extracted text from the image is mapped to sequences of characters. As a reflection of this focus, the main international conference in the area of document analysis, the International Conference on Document Analysis and Recognition (ICDAR) has organised a series of competitions, the *ICDAR Robust Reading Competitions* [29, 37, 38, 71]. With subtasks focused on each one of these stages and publicly accessible datasets and ground-truth data, they have become the standard baseline against which to compare the performance of text detection, extraction or recognition algorithms.

A detailed review of the many publications in this area is beyond the scope of this introduction, but some excellent reviews of the early works are [6, 27, 34] and [85], where more recent approaches are also surveyed in [84] and [79]. However, I would like to highlight some significant state-of-the-art techniques. The work by Epshtein et al. [17] introduced the Stroke-Width

Transform (SWT) that obtains candidate text regions under the assumption that characters have uniform stroke widths. It uses the Canny operator [5] to find edges on the image and then it estimates the width of each stroke around them. Characters are then formed by grouping strokes with similar widths. The SWT obtained state-of-the-art results on the ICDAR competitions, and has been then successfully used and extended by other researchers, for instance in [7, 12, 53, 83]. In particular, Huang et al. [25] proposed an alternate operator: the Stroke Feature Transform (SFT) which introduced additional cues based on pixels colours.

Another family of methods are based on Maximally Stable Extremal Regions (MSERs), and its generalization as Extremal Regions (ER). Originally developed as a method to detect robust image features [41], MSERs respond well to text regions, even in the face of complex backgrounds or irregular lighting conditions. It has been used for license plate detection [14] and first applied to scene text detection by Neumann and Matas [61]. One insight from [61] is to consider all the partial hypothesis from the different stages of the detection and extraction pipeline at once, instead of just cascading the results of each one of the steps. Neumann and Matas have further developed their technique by introducing an improved text grouping scheme [60], and and a more efficient ER filtering [59] that allowed them to attain real-time performance while keeping competitive results on the ICDAR competitions. MSERs have inspired other approaches, such as the ones by Huang et al. [26], who used a Convolutional Neural Network (CNN) as the text region classifier, and notably by Chen et al. [7], who combined the SWT with MSER to improve text detection on blurred images. In this work, we also explore our own text detection technique based on MSERs [45].

The most recent works on scene text analysis have focused on providing end-to-end solutions that are able to detect, extract and recognize the text. In some approaches the output of the text detection stage is sent to an off-the-shelf OCR engine, e.g. [17]. Others attempt to apply text classifiers on the output of the detection and aggregation stages directly. For example, Wang et al. [81] consider words as normal objects where classical object recognition techniques could be applied. Mishra et al. [51] combine bottom-up text detection operators with top-down lexicon based restrictions into a Conditional Random Field (CRF) used for word detection and recognition. While the recognition results are promising, the main limitation of these methods is that the list of words that can be recognized needs to be pre-populated. The work by Neumann and Matas [61] also reports end-to-end recognition by using a character classifier trained on synthetic images. However, modern OCR engines implement many additional techniques besides character classification, and some evidence suggests that an OCR engine still provides a superior recognition ability on properly binarised and rectified images of text [21].

On the area of reading machines, the use of cameras has allowed the reduction of the footprint of these devices. For example, in the iCARE portable reader [24] the camera made document manipulation less awkward than with flat-bed scanners. Chmiel et al. [8] proposed a device comprising glasses with an integrated camera and a DSP-based processing unit which performed the recognition and speech synthesis tasks. However, these devices were directed mainly towards document reading. In the work by Ezaki et al. [18], a text detection software runs on a PDA with a camera and aims to detect scene text in the user's surrounding. Mancas-Thillou et al. [39] built an assistant device specifically aimed at people with visual impairment that also ran on a PDA with a camera. It was a multi-functional device that included the ability to recognize and

communicate several types of inputs: general categories of objects, colour, text in documents, and bank notes. The Classic knfbReader (which stands for *Kurzweil-National Federation of the Blind Reader*) was a commercial device that had a similar design: a PDA with an OCR software attached to a compact camera. More recently, mobile phones have further increased the portability of reading machines. Several systems have been reported that run on smartphones, e.g. [4, 16, 64, 71], and commercial products exist too, such as the latest version of the knfbReader,[1] Google Goggles, or WordLens[2] (acquired by Google in 2014).

However, little attention has been put into the reading problem, i.e., what to do when the text has been detected or recognized. Even if we had a perfect text recognition system (i.e., a system that for every frame would output the list of all the texts present in the image, as unicode character sequences), that would be hardly enough for an assistant device. Current state-of-the-art scene text understanding methods lack temporal scene awareness. They treat their input as a succession of unrelated images, attempting to segment and recognise the text in them without taking advantage of the fact that the same text is invariably repeated across many consecutive frames in a video sequence. Contrary to the degree of attention enjoyed by text detection, extraction and recognition on single images, text tracking has hardly been investigated considering its importance for a reasonable user interaction in any text detection system involving ego or object motion.

Particle filtering was used by Tanaka and Goto [22, 75] and by Minetto et al. [48] for text tracking. The former works described a wearable system for the blind where text was detected using DCT-based features (on prior works by Goto and his co-workers [72, 74], a simple tracking system was described based on block matching between frames). Tracking was performed by generating particles on candidate text regions in new frames, and they were weighted according to a similarity function between the regions based on cumulative histograms. No perspective correction was performed, and only region identity and limited 2D motion was maintained by this method.

In Minetto et al. [48], candidate text regions were initially segmented using a morphological operator applied at different scales, then classified by means of a Support Vector Machine, and grouped together based on their relative distances and sizes. For each text region, particles were propagated in subsequent frames using a first order motion model. Particles' weights were proportional to a similarity coefficient between the histogram of oriented gradients (HOG) descriptors of the respective image regions.

A different approach was used by Na and Wen [56] by tracking text directly using SIFT Lowe [36]. A global motion between frames, modelled as a similarity transformation [23] was computed by minimizing the least squares distance between the SIFT feature matches. In their work, the authors did not specify how text regions were segmented. Furthermore, the computational complexity of extracting SIFT descriptors from every frame precludes the real-time use of this technique, and the simple motion model limits the usefulness of the method when applied to outdoor hand-held camera scenarios. SIFT was also used in the work by Phan et al. [65] to track and align text regions appearing in a sequence of frames. An integration of the aligned text probability maps was then used to improve OCR accuracy. Their algorithm looks for text

[1] `http://www.knfbreader.com`
[2] `http://questvisual.com/`

in adjacent frames either forwards or backwards, necessarily making it an offline process and unsuitable for real-time operation. Additionally, the method requires a manual selection of the initial text bounding box.

1.2. Our work: a natural scenes text reading machine

The general objective of the present work is to develop a text reading assistant that is automatic, autonomous, interactive and operates continuously on video sequences. The ultimate goal is to construct a system that behaves as closely as possible as if a human interpreter was reading the text to the user. It would acquire video images from a camera and selectively translate the texts in the surrounding environment into spoken phrases.

This work is focused first and foremost on video, and the aim of this research has been to exploit the fundamental difference between a video sequence and a high resolution still image: the temporal redundancy vs. the spatial resolution. The key insight of this approach is to consider the input, not as a sequence of unrelated static images, but as a continuously changing view of the world. This is achieved by tracking text regions, maintaining their identity across the video sequence. In addition to the scene awareness, it also enables the utilisation of complex text recognition algorithms in real-time: a fully fledged OCR engine can be run in the background on the detected text regions while the real-time tracking keeps the region identity. When the text recognition results are available, they can be linked to the tracked region, which also helps in fulfilling another requisite for such a device, which is real-time operation. Intuitively, this recognition-while-tracking mechanism is much closer to how humans read text: we do not look at the world, memorise a static image and try to extract all the text in it, but we rather look for text and read it sequentially while our brain is continuously tracking the environment. Tracking would also provide the needed context awareness for an assistance device. For example, it would know when a text has been already recognised with enough confidence or which texts have been read back to the user, in order to not repeat them more than once. If the user is approaching a text to see it better, or to avoid an obstruction or reflection, the system would realize that a piece of text (that was previously unrecognisable) has now become readable. Image registration of multiple frames can also be built on top of a tracking framework, for example, to achieve super-resolution text as a preprocessing stage before OCR. This has been attempted before but on single images [40].

In the rest of this section, I will outline the articles published during the course of this PhD and the context in which they were produced. Of those, [46, 67] and [44] are the journal papers that conform the PhD by publication, while [43] and [45] were presented at conference workshops, but also cover significant parts of the research carried out during the PhD.

In our first approach [43]*, we presented a near real-time scene text tracking system based on Particle Filtering (PF). This work also presented the first iteration of our Adaptive Thresholding based text detection algorithm, and the text aggregation technique based on Delaunay graphs and saliency filtering, which was aimed at forming text entities (i.e. words or groups of words forming small sentences). Each text entity was assigned a Particle Filter (PF) tracker which

*[43] C. Merino and M. Mirmehdi. 'A Framework Towards Realtime Detection and Tracking of Text'. In: *Camera Based Document Analysis and Recognition*. 2007, pp. 10–17. The full text is available on Appendix A.

maintained a set of features and a simple state (a 2D translation and an in-plane rotation). The particles' weights were computed as the number of matched features within a search area, where the identity of individual features was established using SIFT [36] descriptors. A key aspect of this method was the use of just a few high quality features for tracking – in this case, segmented characters. This required a full text segmentation stage per frame (and thus demanded a very fast text segmentation algorithm), but the advantage was that the trackers were more resilient to big changes in orientation, occlusions and illumination changes. This early work, although very useful as a proof of concept, was found to have certain drawbacks where some performance improvement was necessary. For instance, over 80% of the frame processing rate was associated with feature matching, i.e. SIFT, which is a computationally expensive operator. Additionally, the number of SIFT descriptors produced by each feature (i.e. characters) was rather low, and limited the ability to discriminate between measurements. SIFT is affine invariant but not perspective invariant [47], and no estimation of the 3D spatial orientation of the text in the scene was performed so the whole system was sensitive to wide baseline changes. This is also related to the simple state model used, namely just a 2D translation and in-plane rotation, that traded accuracy and robustness in favour of low computational complexity and hence better frame rate.

To overcome the limitations of our PF and SIFT based text tracker, I started exploring the use of a different region matching technique. In this new approach image regions were matched using just Normalised Cross Correlation (NCC), but the region extraction was coupled with the state maintained by the tracking filter, thus being covariant with the state changes. The tracker still used a simple 2D translation, scale and in-plane rotation state model (which also limited the region matching to being covariant to the similarity transformation), but otherwise it was a major rewriting of our tracking framework, as the objective was also to increase its performance by making use of several processing cores in parallel. In this iteration, the system was also capable of performing text recognition using an off-the-shelf OCR engine. The ongoing progress of this effort was reported in [67]*. We still faced a fundamental challenge regarding our tracking approach: the simple state model precluded its use in any but the most simple scene text scenarios.

A concurrent work to the purely computer vision efforts was the construction of a reading machine prototype. In [45]† we presented a wearable head-mounted device that encased a camera inside a flat-cap. This is the *reading hat*, our prototype text reading machine. In this work we also explored the use of MSERs as a distinguished text region detector, and introduced Hierarchical MSER, a technique to efficiently prune a component tree of detected text regions. We presented comparative text detection results against the ICDAR 2003 Robust Reading Competition image database [37] which demonstrated the speed of our system, as measured against the published state-of-the-art, although our precision and recall values were still inferior to the best performing works.

*[67] A. Rodríguez-Hernández, C. Merino, O. Casanova, C. Modroño, M. Torres, R. Montserrat, G. Navarrete, E. Burunat and J. González-Mora. 'Sensory substitution for visually disabled people: Computer solutions'. In: *WSEAS Transactions on Biology and Biomedicine* 7.1 (2010), pp. 1–10. The full text is available on Appendix B.

†[45] C. Merino-Gracia, K. Lenc and M. Mirmehdi. 'A Head-Mounted Device for Recognizing Text in Natural Scenes'. In: *Camera Based Document Analysis and Recognition*. Ed. by M. Iwamura and F. Shafait. Vol. 7139. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2012, pp. 29–41. The full text is available on Appendix C.

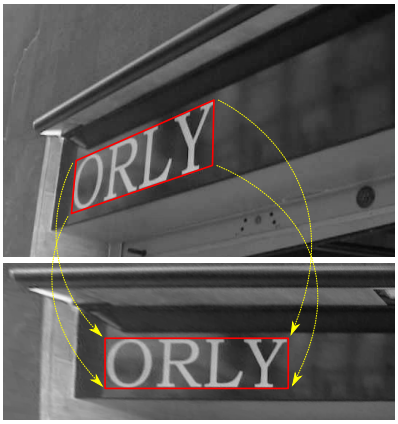


Figure 1.2. A projective transformation of text. A rectangle enclosing the text is seen as a quadrilateral in the source image (a) and is mapped into a rectangle in the target image (b).

Once it became apparent that the major limitation of our text tracker so far was the limited state model, I needed to reconsider the problem of text extraction, and in particular, the correction of text perspective distortions prior to the text tracking stage. Assuming text lies on a planar surface, the process of perspective recovery of text can be modelled as a projective transformation [23] 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 Figure 1.2). This projective transformation or homography is represented by a 3×3 mapping matrix:

$$\mathbf{p}' = \mathbf{H}\,\mathbf{p}\ , \tag{1.1}$$

where $\mathbf{p} = [\,\mathrm{x}\ \mathrm{y}\ 1\,]^\mathsf{T}$ and $\mathbf{p}' = [\,cx'\ cy'\ c\,]^\mathsf{T}$ 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.

The most relevant prior work specifically dealing with 3D scene text recovery was the one by Myers et al. [54] whose method deals with individual or isolated text lines found in everyday scenes, particularly outdoors. In that work, 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.

Myers et al. [54] pointed out that some of the homography 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 could 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 [11]. Correcting shear and vertical foreshortening is a challenging problem due to the difficulty of obtaining accurate vertical cues for text [9] – even more so when only one text line is being considered. Myers et al.'s [54] 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. [54], 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.

In [46]*, we presented a novel technique for efficient perspective estimation of text which significantly improved the existing state-of-the-art in both accuracy in terms of ranges of angles and speed. 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. We also presented systematic comparative results that showed the improved recognition accuracy across a larger range of angles against the work by Myers et al. [54]. An important aspect of this work, that sets it apart from most other text rectification techniques in the literature, is the full homography estimation, which is crucial in order to be able to use it as the state of a tracking filter.

In our most recent work [44]†, I revisited the text tracking problem by looking at performing perspective aware tracking, built on top of our previous works in the areas of text detection, aggregation and perspective estimation. Each text region is independently tracked by an Unscented Kalman Filter (UKF) that keeps an homography transformation of the text into a fronto-parallel view. Tracking is performed on high level features (i.e. perspective corrected characters). Again, this provides several advantages over low-level tracking of feature points, such as increased resiliency against orientation changes and occlusions. Tracking allows *(i)* to maintain region identity across the sequence, *(ii)* to smooth the estimation of the region's parameters (position and 3D orientation) to reduce jitter. Both of these outcomes play a major role in facilitating scene awareness, in reduction of false positive segmentations and increase in recognition accuracy, and in better interactive communication of text in the environment to the user, e.g. by managing the frequency of communicating text seen in the scene as an audio signal to a blind user.

*[46] C. Merino-Gracia, M. Mirmehdi, J. Sigut and J. L. González-Mora. 'Fast perspective recovery of text in natural scenes'. In: *Image and Vision Computing* 31.10 (2013), pp. 714–724. The full text is available on Appendix D.

†[44] C. Merino-Gracia and M. Mirmehdi. 'Real-time text tracking in natural scenes'. In: *IET Computer Vision* 8.6 (2014), pp. 670–681. The full text is available on Appendix E.

The prototype presented in this work operates in real-time and it is autonomous, i.e. new trackers are created when new text entities are found and their identity is kept for as long they are in view; they are removed when the entities are no longer detected, given some resilience to brief occlusions. Trackers are automatically selected for OCR – a process carried out in parallel to tracking by an off-the-shelf OCR engine on the perspective corrected image patches extracted from the input sequence. This demonstrates the role tracking plays, i.e. when a text recognition result is available, the system is able to relate it to the original text entity even if the camera has moved. Likewise, available recognition results are automatically selected for synthesising into audio (using text-to-speech) and are played back to the user.

1.3. Summary

This chapter has described the problem, context and focus of this PhD. It has also outlined the motivations and contributions of the publications produced as a result of this research. Next, Chapter 2 will describe in detail the text reading system developed over the course of this PhD.

Chapter 2

Description of the reading system

This chapter describes the proposed end-to-end text reading system and presents our wearable text reading prototype. It is a summary of the methodology of the works presented as part of this PhD by publication [43–46, 67] (the full text of these articles are included in Appendices A to E). To give a general overview, the proposed end-to-end real-time text reading system is illustrated in Figure 2.1. First, video images are acquired from a camera and the text regions are detected. Section 2.1 explains our text detection technique and, in particular, Sections 2.1.1 and 2.1.2 introduce two text segmentation methods based on Adaptive Thresholding (AT) and Maximally Stable Extremal Regions (MSERs) respectively. Then, text regions are aggregated to form lines of text. A method for efficient multi-orientation perceptual text aggregation based on Delaunay graphs is presented in Section 2.2. Next, the orientation of each line of text is estimated. A fast method for perspective text rectification in natural scenes is described in Section 2.3. Finally, text regions and their 3D orientations are tracked frame-to-frame using Unscented Kalman Filters (UKFs), a process which is explained in Section 2.4. Those stages represent the real-time components of our text reading prototype. Concurrently to them, Optical Character Recognition (OCR) is performed on the tracked regions and speech synthesis on the recognized text, a process that is covered in Section 2.5. To conclude the chapter, the *reading hat*, our prototype text reading machine, is described in Section 2.6.

2.1. Text detection and segmentation

The first stage in our text reading system is text detection, in which each input image is segmented to obtain a set of candidate text regions, containing possibly none, or one, or more characters. Later, these regions will be used as measurements for the tracking filter, as well as serving as the building blocks for text line aggregation and tracker creation. Our segmentation algorithm was first presented in [43], and an alternate region detection mechanism based on MSER was later presented in [45]. They are described in Sections 2.1.1 and 2.1.2 respectively.

2.1.1. Adaptive thresholding and hirerarchical filtering

Adaptive Thresholding (AT) is applied to binarize the input image and retrieve a set of Connected Component (CC) regions. A tree is then constructed to represent the topological relationship between these CCs (see Figure 2.2). A key step of this algorithm is the hierarchical

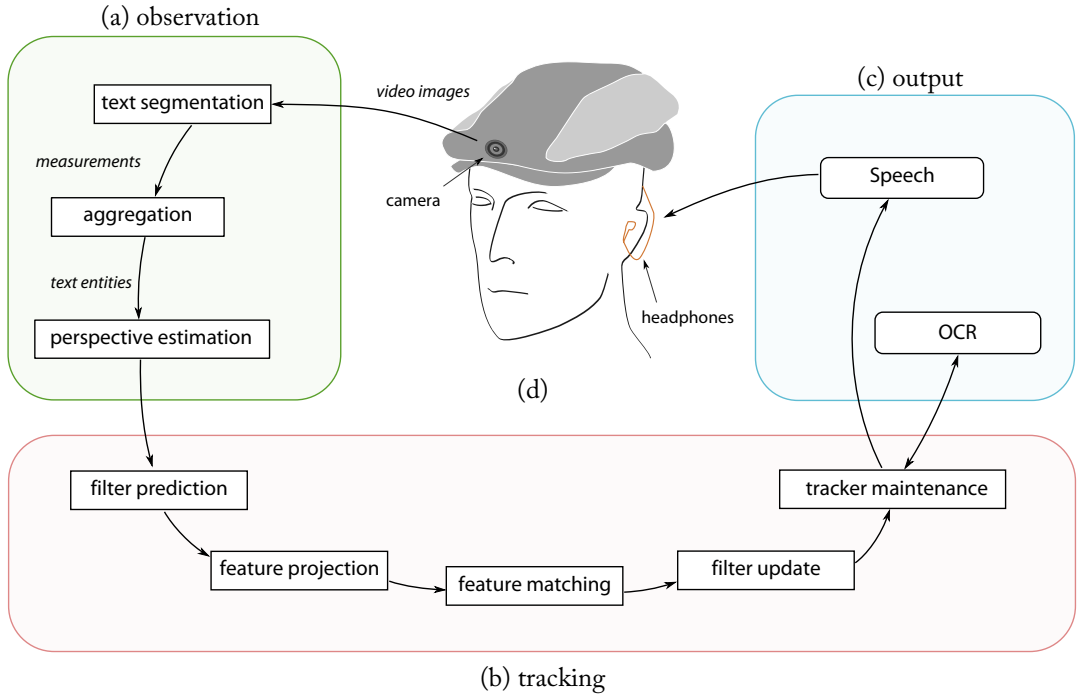


Figure 2.1. A schematic of the proposed end-to-end real-time text reading system. Video images are captured and the text is first segmented and aggregated to form lines of text (a). Then, text regions are tracked from frame to frame (b). Each tracked region will independently be run through OCR and selectively synthesized into speech (c). These are played back to the user, who carries a wearable reading hat (d).

filtering of the tree nodes, based on the assumption that in natural scene images with text, structural elements (such as sign borders, posters frames etc.) can be discarded purely based on their hierarchical relationships with other text regions. The filtering works as follows. Once the tree is built, it is walked depth-first and each node of the tree is classified as text or non-text during the walk (Figure 2.3a) using a cascade of text region filters as described later below. When a node has children already classified as text, it is discarded as non-text, despite the text classifying functions might have marked it as text. This discards most of the non-text structural elements of the text (Figure 2.3b). Additionally, the tree filtering approach allows for the segmentation of dark and light text *in one pass only*. Figure 2.6a shows an example image and Figure 2.6b illustrates the corresponding CCs (i.e. candidate text regions) detected at this stage.

To classify text regions we apply a series of tests in cascade, meaning that if a test discards a region as non-text, no more tests are applied to it. Using a small number of tests is important for real time processing; coarse but computationally more efficient tests are applied first, quickly discarding obvious non-text regions, and slower, more discriminative tests are applied last, where the number of remaining regions is fewer. The tests we apply are on size, aspect ratio, border energy and an eigenvector based texture measure adapted from [77].

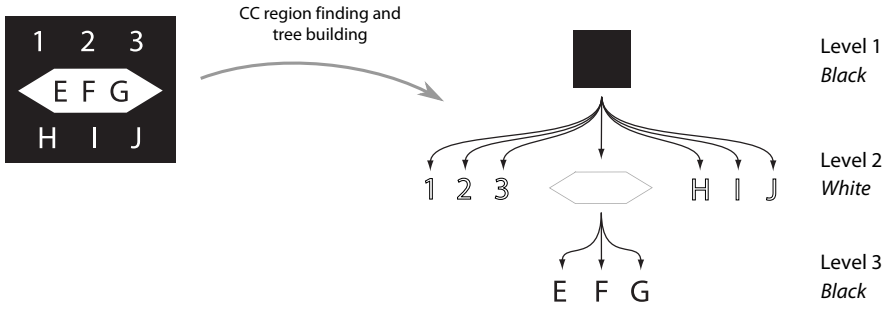


Figure 2.2. A synthetic sample image and its corresponding tree of connected regions.

Size and aspect ratio Regions too big or too small are discarded. The thresholds here are set to the very extreme. Very small regions are discarded to avoid noise. This may still drop characters, but they probably would be otherwise impossible to recognise by OCR and as the user gets closer, they are more likely to be picked up anyway. Large regions are discarded because it is unlikely that a single character occupies very large areas (over half the size) of the image. Additionally, regions that have really extreme aspect ratio (i.e., they are either too tall or too wide) are also filtered.

Border Energy This is a measure of contrast against the background. It filters out regions with low average edge response (from a Sobel operator (S_x, S_y)) around its boundary set of points $\mathbf{B}$, i.e., the region is valid only if its border energy exceeds a threshold e:

$$\frac{1}{|\mathbf{B}|} \sum_{(x,y)\in\mathbf{B}} \sqrt{(S_x(x,y)^2 + S_y(x,y)^2)} > e \ . \tag{2.1}$$

This removes regions that usually appear in less textured and more homogeneous regions.

Texture measure The last filter in the sequence is a measurement of texture response, as text regions usually contain high frequencies [50]. We found that the LU transform [77] yields good response results when applied to text regions. It is a simple transformation based on the LU decomposition of square image sub-matrices A around each interest point.

$$A = P\ L\ U \ , \tag{2.2}$$

where L and U are lower and upper diagonal matrices and the diagonal elements of L are equal to one. Matrix P is a permutation matrix. In the LU decomposition, the number of zero diagonal elements of U is in direct proportion to the dimensionality of the null-space of A.

The actual texture response $\mathsf{LU}(l, w)$ is calculated as the mean value of the diagonal values

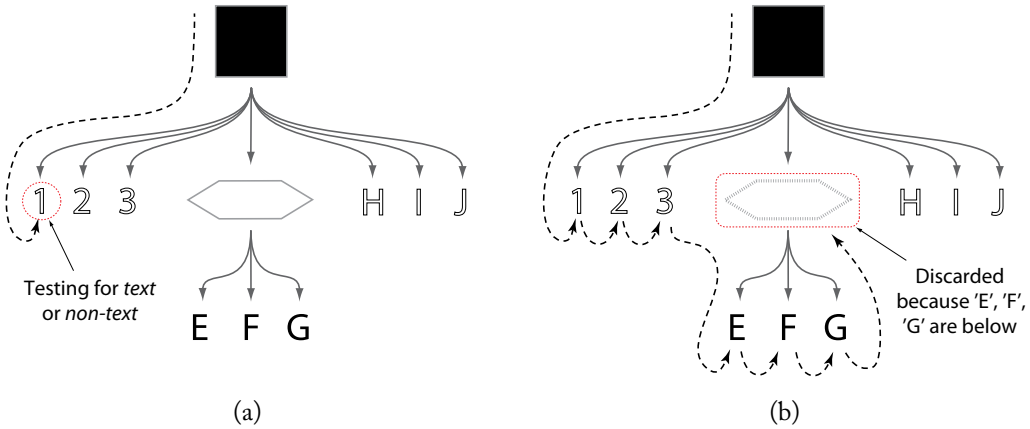


Figure 2.3. Depth first CC tree walking. (a) During the walk, each leaf node is classified as *text* or *non text*. (b) Parent nodes are discarded when children are classified as text.

of the U matrix.

$$\mathsf{LU}(l,w) = \frac{1}{w-l+1}\sum_{k=l}^{w}|u_{kk}|\ , \quad 1<l<w\ , \tag{2.3}$$

where w is the window size and l number of skipped lower frequency values. The texture response for a candidate component is calculated as the mean LU transform value of a sampled set of points ($\mathbf{N}$) inside the bounding box of the region. The region is considered text if the response is above a certain threshold t.

$$\frac{1}{|\mathbf{N}|}\sum_{p\in\mathbf{N}}\mathsf{LU}(l,w) > t\ . \tag{2.4}$$

In the filters above, the thresholds were determined empirically and fixed in all our experiments to: $e = 40$ and $t = 1.9$.

2.1.2. Hierarchical MSER text segmentation

Maximally Stable Extremal Regions (MSERs) [41] are regions of interest in an image which present an extremal property of the intensity function around its contour. When applying a varying threshold level to a grey scale image, Connected Component regions in the thresholded image evolve: new regions appear at certain levels, regions grow and eventually join others. Those regions which keep an almost constant pixel count (area) for a range of threshold levels are called MSERs. This technique, originally proposed as a distinguished region detector [41], also presents very desirable properties when applied to text detection, such as stability and multiscale detection.

MSERs are related to the concept of the 'component tree' [58] of the image, as shown by Donoser and Bischof [13]. The component tree is a representation of all the CCs which result

(a) original (b) hierarchical MSER

Figure 2.4. Hierarchical MSER representation. Blue regions were obtained by the MSER+ pass and the red regions by the MSER- pass. Darker regions represent upper tree nodes (closer to the root), while brighter regions show lower nodes (closer to the leaves).

from applying a varying threshold level to a grey scale image. The CCs are laid out in a hierarchy representing the topological relationship between them. MSERs can be obtained by filtering the component tree: a stability factor – i.e. the rate of change in the area of the components – is computed for each node in the tree. MSERs are identified as local minima of the stability factor along paths in the tree towards the root. The component tree is used here as a tool to efficiently compute the MSERs. However, the hierarchical layout of the MSERs as provided by the component tree can also be exploited to drive a text/non-text region filtering, in a similar way to our previous Adaptive Thresholding based technique (Section 2.1.1).

We use the efficient, linear time MSER algorithm by Nistér and Stewénius [63], which crucially also constructs the component tree. We make two passes on the original image. First, MSER+ regions are obtained by applying the MSER algorithm on the image. This produces light regions inside dark ones. Then MSER- regions are obtained by applying the MSER algorithm to the inverse (negative) of the original image which produces dark regions inside light ones. The sets of regions returned by each pass are disjoint and both passes are needed to detect light text on dark backgrounds and dark text on light backgrounds. The algorithm can be easily modified to return a *hierarchical MSER* tree; an example output can be seen in Figure 2.4b, where blue regions were obtained by the MSER+ pass and the red regions by the MSER- pass. Darker regions represent upper tree nodes (closer to the root), while brighter regions show lower nodes (closer to the leaves). With hierarchical MSER, we have the desirable properties of MSERs as a distinguished region finder applied to text detection. Additionally, we keep the topological relationship of the CCs, which provides context information for later text filtering stages.

The resulting hierarchical MSER tree is then pruned in two stages: *(i)* reduction of linear segments and *(ii)* hierarchical filtering. The first stage identifies all the linear segments within the tree where a linear segment is a maximum path between two tree nodes without any branches in between. Likewise, it is a path starting with a node with only one child, and ending with a branch node (a node with more than one child), or a leaf. Each linear segment is then collapsed into the node along the path, as shown in Figure 2.5, which maximizes the border energy function

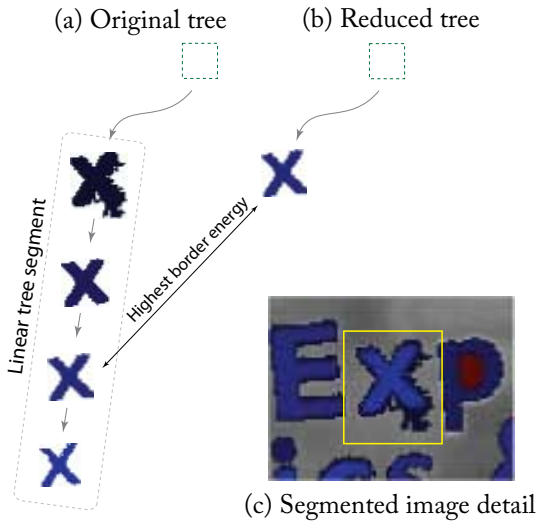


Figure 2.5. Linear tree segments removal.

(see Equation 2.1). In the second stage, the tree is walked depth-first in a similar way to the hierarchical tree filtering proposed in Section 2.1.1, and the filters applied are the same: size, aspect ratio, border energy and texture measure.

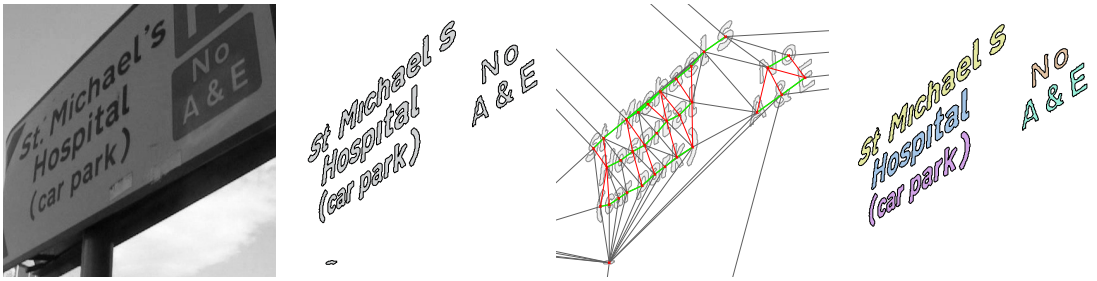
2.2. Text aggregation

In order to be able to extract common clues for perspective estimation, for tracking and for the purpose of text recognition by (off-the-shelf) OCR, we need to group the CC regions together to form text entities. The grouping is performed by first determining which CC regions are closely associated by evaluating a visual saliency measure between each pair of regions, and then by searching for dominant orientations to separate independent lines of text. The saliency filtering technique (Section 2.2.1) was first presented in [43] and later refined with the addition of histogram filtering (Section 2.2.2) in [46].

2.2.1. Saliency filtering

First, a Delaunay triangulation [1] to join 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 2.6c 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 [66].

We use the two saliency operators introduced by Pilu [66]: the *blob dimension ratio* (BDR) and the *relative minimum distance* (RMD). Given two CC regions, $\mathcal{A}$ and $\mathcal{B}$, BDR evaluates the



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

Figure 2.6. 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.

similarity in size between them, i.e.

$$\mathsf{BDR}(\mathcal{A},\mathcal{B}) = \frac{\mathcal{A}_{\min}+\mathcal{A}_{\max}}{\mathcal{B}_{\min}+\mathcal{B}_{\max}}\ , \tag{2.5}$$

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.

$$\mathsf{RMD}(\mathcal{A},\mathcal{B}) = \frac{D_{\min}}{\mathcal{A}_{\min}+\mathcal{B}_{\min}}\ , \tag{2.6}$$

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, which can be efficiently computed from the region's convex hull. The combined saliency operator between the two text regions is then:

$$\mathbf{P}(\mathcal{A},\mathcal{B}) = N(\mathsf{BDR}(\mathcal{A},\mathcal{B}),1,2)\cdot N(\mathsf{RMD}(\mathcal{A},\mathcal{B}),0,4)\ , \tag{2.7}$$

where $N(x,\mu,\sigma)$ is a Gaussian distribution with mean μ and standard deviation σ (the parameters were determined experimentally by Pilu [66]). Edges with $\mathbf{P}(\mathcal{A},\mathcal{B}) < 0.9$ are removed from the graph. In Figure 2.6c, edges removed during the saliency filtering are represented in grey.

2.2.2. Histogram filtering

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

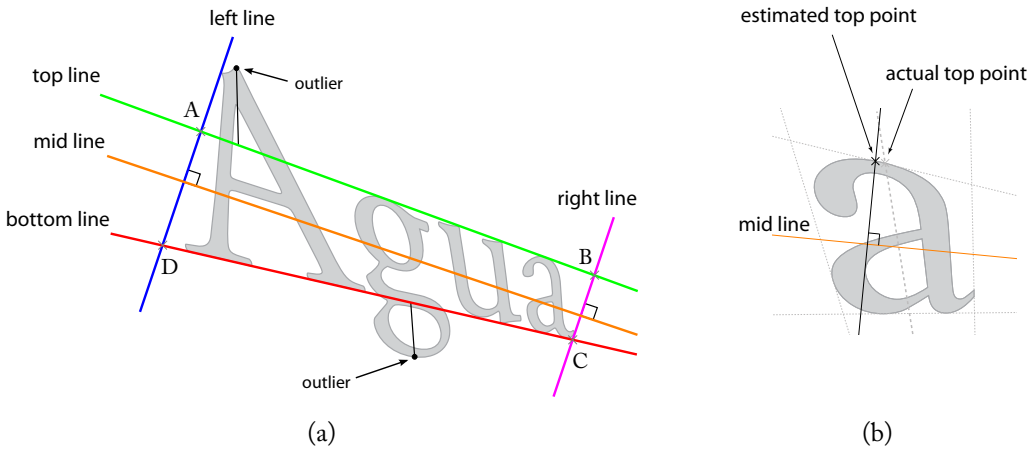


Figure 2.7. (a) Top, mid, bottom, left and right lines along with the formed quadrilateral. The outliers of the line estimation are also illustrated here. (b) Top point estimation – on severely perspective distorted characters the estimated top point and the actual top point might not correspond.

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 Figure 2.6c, filtered edges at this stage are represented in red, and the remaining connected subgraphs are represented in green. Finally, Figure 2.6d 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 next stage, which is 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 chapter, the unit of work is the text line.

2.3. Perspective estimation

At this stage of text recovery, we estimate a 3×3 homography matrix transformation of a text entity into a fronto-parallel representation. This technique was first introduced in [46]. Assuming a pinhole camera model, the homography is the transformation that allows the modelling of all possible orientation changes the text can undergo in an image without the need to have calibrated cameras or an explicit 3D representation. The ability to quickly estimate the text orientation makes high-level perspective-aware tracking possible, as well as the extraction of perspective invariant feature descriptors.

The orientation detection is performed in two steps: parallel rectification and shear estimation.

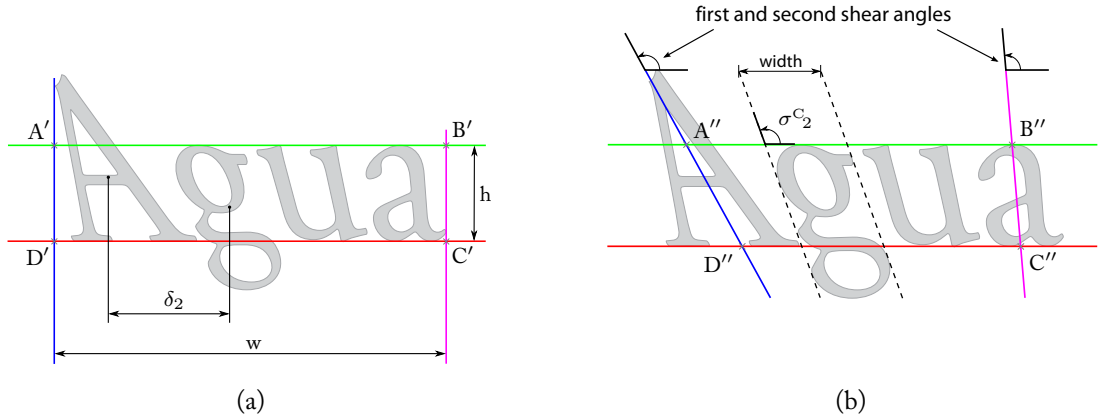


Figure 2.8. (a) Image partially rectified according to the top, bottom, left and right lines. The displacement (δ_2) for the second character is also shown. (b) Quadrilateral formed after computing the two shear angles. Additionally, the *upright shear angle* (σ^{C}) for the second character is shown.

2.3.1. Parallel rectification

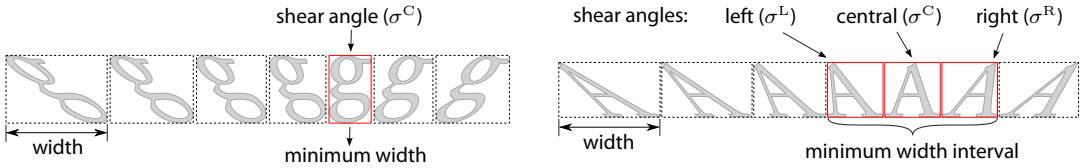
A line is fitted to the centre point of every character in the text line (the centre of mass already computed before), using a least squares method, and named the *mid-line*. As used and defined here, this line will not usually correspond to any conventional typography line in the text. Possible errors 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 Figure 2.7b). 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 [19] 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 Figure 2.7a).

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 Figure 2.7a.

A straightforward homography $\mathbf{H}_p$ from four pairs of matching points [23] is computed so that the quadrilateral (ABCD; Figure 2.7a) is mapped to a rectangle (A′B′C′D′; Figure 2.8a). The aspect ratio and size of the target rectangle are still unknown, but not significant as the OCR



(a) one shear angle estimation (b) three shear angles estimation

Figure 2.9. Shear estimation for one character.

engine is scale 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:

$$\mathrm{w} = \max(d(\mathrm{A}, \mathrm{B}), d(\mathrm{C}, \mathrm{D})) \ , \tag{2.8}$$

$$\mathrm{h} = \max(d(\mathrm{A}, \mathrm{D}), d(\mathrm{B}, \mathrm{C})) \ , \tag{2.9}$$

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*.

2.3.2. 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 Figure 2.8a). 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 ($\delta_1 = 0$). Figure 2.8a 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 Figure 2.9a), however, some characters have a range of angles, e.g. those with a triangular shape such as letters 'A' or 'V' (see Figure 2.9b). 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,

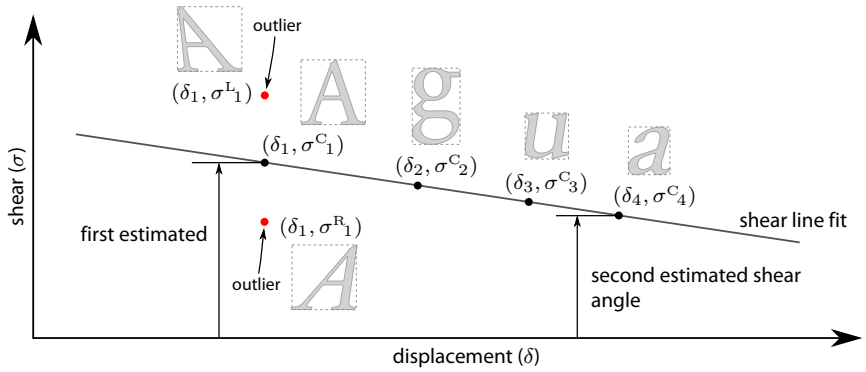


Figure 2.10. Shear angles estimation. From the alternative shear candidates of the first letter ($\sigma^{L}{}_0$, $\sigma^{C}{}_0$ and $\sigma^{R}{}_0$) the wrong ones (in red) are discarded as outliers by the RANSAC line fitting.

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 Figure 2.10 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 Figure 2.10).

On an implementational note, the upright shear angle can be efficiently computed using a variation of the Rotating Calipers paradigm [78]. 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 Figure 2.8b, the upright shear angle for the second character (i.e. $\sigma^{C}{}_2$) 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 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 Figure 2.8b). A homography $\mathbf{H}_s$ mapping this new quadrilateral to a rectangle is computed. The result of this transformation is the rectified image.

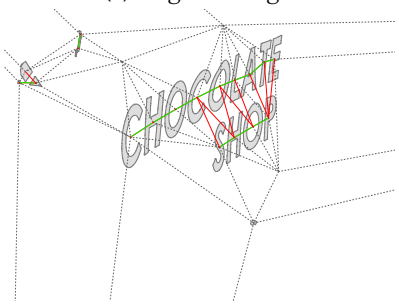
Thus, the full rectifying homography for the original image is the combination of both partial rectifications:

$$\mathbf{H} = \mathbf{H}_s \, \mathbf{H}_p \; . \tag{2.10}$$

As a summary so far, Figure 2.11 illustrates the end-to-end text detection, grouping and perspective rectification stages on an example image.

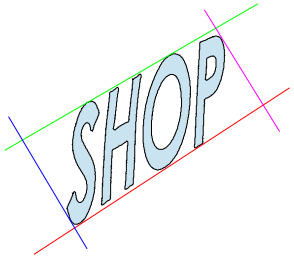
(a) original image

(b) segmented components

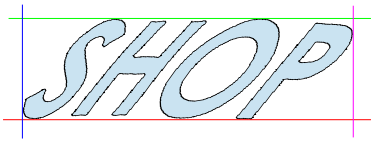


(c) association graph

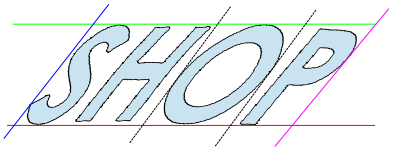
(d) grouped text lines



(e) top/bottom line estimation



(f) parallel image



(g) shear estimation

(h) full rectification

Figure 2.11. Text segmentation and perspective estimation for an example image (a). First, for the segmentation and aggregation stages: (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. Then, for the perspective estimation step: (e) the top and bottom lines estimation, (f) the parallel image and (g) the shear estimation. Finally, (h) the full perspective rectification.

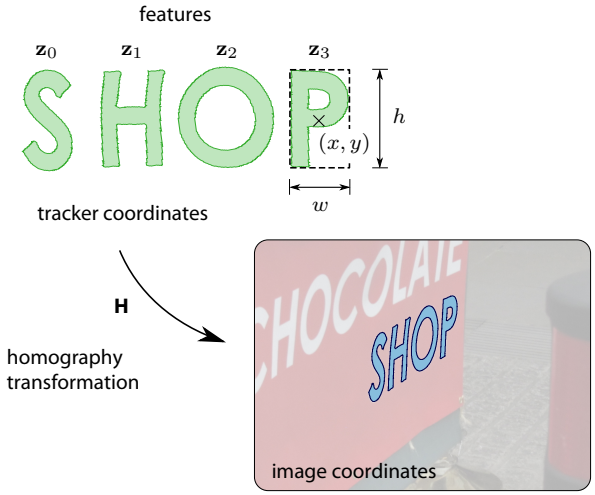


Figure 2.12. Tracker representation, which keeps a set of features $\mathbf{z}_i$ and the state of the UKF that produces the homography $\mathbf{H}$.

2.4. Text tracking

We now describe our proposed text tracking approach, which was first presented in [44]. Once a text entity is identified, a tracker is created to follow the text region from frame to frame while it is in camera view. The detailed process of tracker creation and removal is explained later in Section 2.4.1. For now, for ease of exposition, we assume that a set of trackers already exists and properly initialized to follow a corresponding set of text entities in the scene.

A tracker is characterized by a set of tracked features $\mathbf{z}_i$ and a dynamic state $\mathbf{x}_k$, which is updated by a predictive filter. The features $\mathbf{z}_i$ correspond to the individual characters in the text line and are used as the anchor points to be matched against image measurements during the observation stage of the filter. They are stored in a fronto-parallel representation, in a coordinate frame referred to as *tracker coordinates* (see Figure 2.12). Each feature is defined as

$$\mathbf{z}_i = \begin{bmatrix} x & y & w & h \end{bmatrix}^\mathsf{T} , \qquad i = 1, \dots M , \tag{2.11}$$

where (x, y) is a feature's centre point, (w, h) are the dimensions of the feature's bounding box, and M is the number of features. Additionally, each feature keeps a perspective corrected image patch used during feature matching (as seen in Figure 2.12).

As previously outlined in Chapter 1, our first approach [43] used Particle Filtering [15] since they model a non-linear system, such as our text tracking problem, well. However, in this work we explore the use of the Unscented Kalman Filter (UKF) [80] for tracking because it provides the uncertainty of the system's state estimation via a Gaussian probability distribution, and it is also more efficient, i.e., to achieve the same accuracy as the PF, it needs to use substantially fewer sampling points. The UKF is derivative free and employs a deterministic sampling approach (the Unscented Transform, UT) to propagate the density function across non-linear state changes.

The UT captures the non-linearities better than alternatives such as the Extended Kalman Filter and is easier to implement.

We represent the filter state by a homography transformation $\mathbf{H}$ mapping the fronto-parallel view of the tracked features in tracker coordinates to image coordinates (Figure 2.12). The homography can be characterized by a vector $\mathbf{h}$:

$$\mathbf{h} = \begin{bmatrix} t_x & t_y & \theta & s_x & s_y & \sigma & l_x & l_y \end{bmatrix}^\intercal , \tag{2.12}$$

where (t_x, t_y) defines a translation, θ is an in-plane rotation angle, (s_x, s_y) is an anisotropic scale, σ is a shearing, and (l_x, l_y) is a foreshortening around both axes. Given the homography, there is a closed form unique solution for all the parameters [23] (refer to [44, , Appendix] for a formulation of this solution) which we name, along with its inverse, the *homography (de)composition function* $\mathcal{H}$:

$$\mathbf{H} = \mathcal{H}(\mathbf{h}) , \tag{2.13}$$

$$\mathbf{h} = \mathcal{H}^{-1}(\mathbf{H}) . \tag{2.14}$$

Although both representations are mathematically equivalent, with this decomposition the filter deals directly with the underlying parameters that define the transformation, enabling the direct estimation of the uncertainty in each parameter via the covariance matrices. The dynamic model also benefits from this representation as we can define velocity vectors that affect only the translation or rotation parameters of the transformation. For the rest of this section, when we refer to the homography $\mathbf{H}$, an implicit conversion will be assumed from the parameters vector $\mathbf{h}$ to the homography using $\mathcal{H}(\mathbf{h})$. The only moment in which the inverse operation $\mathcal{H}^{-1}(\mathbf{H})$ is needed is for tracker creation, as explained in Section 2.4.1.

For the prediction stage of the filter, we use a constant velocity model, where we only consider in-plane translational and angular velocities. We define the velocity vector $\mathbf{v}$ as

$$\mathbf{v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}^\intercal , \tag{2.15}$$

and the state vector of the filter at frame k as a stacking of the homography parameters and velocity vectors

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{h} \\ \mathbf{v} \end{bmatrix} . \tag{2.16}$$

The new state prediction is then

$$\hat{\mathbf{x}}_{k+1} = \begin{bmatrix} \mathbf{h} + \overset{\circ}{\mathbf{v}}\, \Delta t \\ \mathbf{v} \end{bmatrix} , \tag{2.17}$$

where $\overset{\circ}{\mathbf{v}} = \begin{bmatrix} \mathbf{v}^\intercal & \mathbf{0}^\intercal \end{bmatrix}^\intercal$ is the velocity vector padded with zeros to the length of $\mathbf{h}$ and Δt is the elapsed time since the last frame.

The measurement function maps the tracked features to observable characteristics in the image

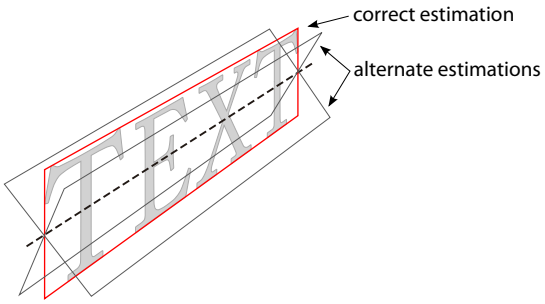


Figure 2.13. Homography estimation ambiguity if only the centre points of each character are considered.

(called *measurements*) using the filter state. However, as each tracker represents a line of text, the centre points of all the characters are roughly aligned. If the centre points were the only points of our measurement function, there would be a great deal of uncertainty for rotations around the horizontal axis (i.e. *elevation* – see Figure 2.13 as an example). Since we have a good estimate of the text orientation, and we know that all the points of a text line lie on a plane, our observation model includes five points per tracked feature $\mathbf{z}_i$: the centre point $\hat{\mathbf{c}}_0$ and the four corner points $\hat{\mathbf{c}}_j$, $j = 1, \ldots 4$ of the feature's bounding box (as shown in Figure 2.14):

$$\hat{\mathbf{z}}_i = \left[\begin{array}{ccccc} \hat{\mathbf{c}}_0{}^\mathsf{T} & \hat{\mathbf{c}}_1{}^\mathsf{T} & \hat{\mathbf{c}}_2{}^\mathsf{T} & \hat{\mathbf{c}}_3{}^\mathsf{T} & \hat{\mathbf{c}}_4{}^\mathsf{T} \end{array} \right]^\mathsf{T} , \qquad i = 1, \ldots M . \tag{2.18}$$

These are converted from tracker coordinates using the predicted state homography. For example, $\hat{\mathbf{c}}_0$ is computed as the transformation of (x, y) to image coordinates, $\hat{\mathbf{c}}_1$ as the transformation of $(x - w/2, y - h/2)$, and likewise for $\mathbf{c}_2$ to $\mathbf{c}_4$.

The predicted observation is then a combination of all of the individual feature mappings:

$$\hat{\mathbf{y}}_k = \left[\begin{array}{ccc} \hat{\mathbf{z}}_0{}^\mathsf{T} & \cdots & \hat{\mathbf{z}}_M{}^\mathsf{T} \end{array} \right]^\mathsf{T} . \tag{2.19}$$

Finally, as the last part of the observation model, the tracked features need to be matched against the segmented text regions or candidate measurements. To discriminate between the candidates, each feature keeps an image patch, normalized to 50×50 pixels. It is a perspective corrected image patch, extracted using the four corner points of the feature from the frame in which the tracker was created. Matching is performed using NCC and only on measurements within a certain search radius around the predicted feature position. The search radius is obtained from the filter's state covariance matrix, as it represents the uncertainty in the new state's prediction. After matching, each feature has one measurement candidate. As with the observation function, each measurement $\mathbf{m}_i$ is defined by the centre point of the region and its four corner points:

$$\mathbf{m}_i = \left[\begin{array}{ccccc} \mathbf{c}_0{}^\mathsf{T} & \mathbf{c}_1{}^\mathsf{T} & \mathbf{c}_2{}^\mathsf{T} & \mathbf{c}_3{}^\mathsf{T} & \mathbf{c}_4{}^\mathsf{T} \end{array} \right]^\mathsf{T} , \qquad i = 1, \ldots M , \tag{2.20}$$

where $\mathbf{c}_j$, $j = 0, \ldots 4$, are the mapped centre and corner points, and are obtained from the

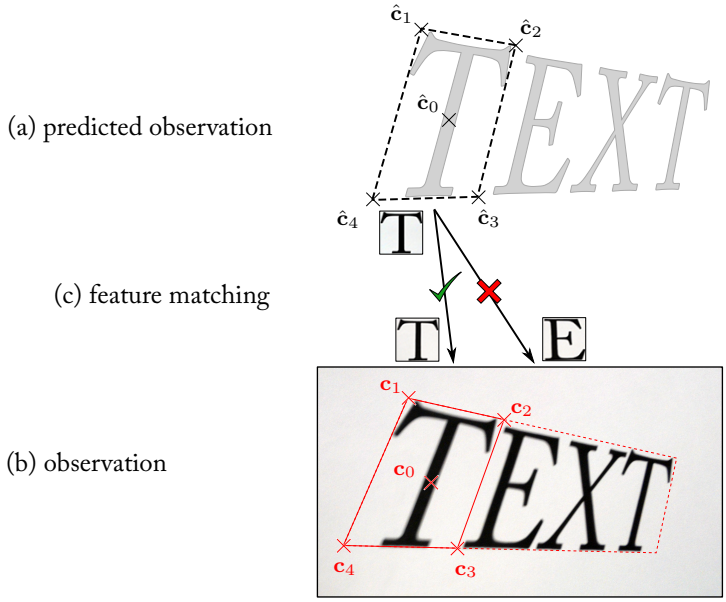


Figure 2.14. The observation model. First, a new location for the tracker features is predicted (a). Here the 5 predicted observation points $\hat{\mathbf{c}}_i$ for the first feature are represented in the upper part. Then, from the segmentation and perspective orientation stage, the actual measurement points $\mathbf{c}_i$ are obtained (b). Finally, the candidate measurements are matched using perspective corrected image patches which are also shown (c).

orientation estimation stage. Hence, the observation used by the UKF is:

$$\mathbf{y}_k = \left[\begin{array}{ccc} \mathbf{m}_0{}^\mathsf{T} & \cdots & \mathbf{m}_M{}^\mathsf{T} \end{array} \right]^\mathsf{T} . \tag{2.21}$$

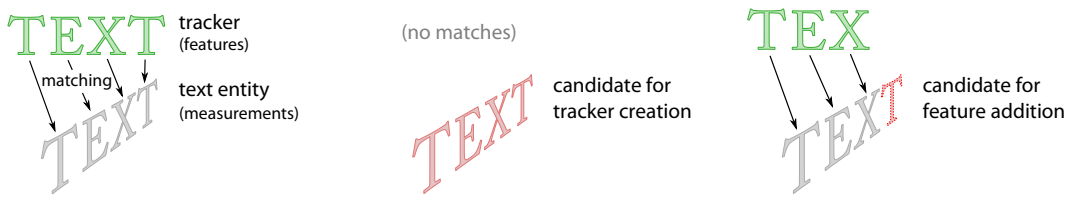
In Figure 2.14 the observation model is illustrated: the predicted observation, the actual observation and the feature matching.

2.4.1. Tracker maintenance

Tracker maintenance refers to the set of mechanisms in which new trackers are created, new features are added to existing trackers, and bad trackers are removed. This allows the automatic continuous operation of the system.

At first we need to correlate the text entities produced in the text association stage to the current set of trackers. The text association stage groups measurements as belonging to the same text entity and the tracking stage may associate features to some of the measurements after matching.

By correlating tracked features to measurements and then to text entities, several possibilities arise: *(i)* a tracker has matched all the measurements belonging to a text entity – this is the perfect tracking case and no further action is needed (Figure 2.15a); *(ii)* no tracker has matched any of the measurements of a given text entity – the entity is then a candidate for *tracker creation*



(a) perfect tracking (b) tracker creation (c) feature addition

Figure 2.15. Tracker maintenance.

(Figure 2.15b); *(iii)* a tracker has matched some of the measurements inside an entity – the remaining (unmatched) measurements are candidates for *feature addition* to that tracker (Figure 2.15c). Additionally, when a tracker matches most or all of its features it is considered a good track. Likewise, if a tracker did not match any of its features (or only matched a low fraction of them), it is considered a bad track or a *mistrack*. After a certain number of frames being a bad track, a tracker is removed. These operations are further explained in the following.

Tracker creation When a text entity does not have any tracker matching any of its features, it is considered an untracked entity, thus requiring a tracker to be created for it. Tracker creation proceeds as follows: the perspective estimation stage returns a homography transformation $\mathbf{H}'$ of the measurements in the group to a fronto-parallel representation. All the measurements are converted into features in the new tracker by applying this homography transformation and then obtaining the centre point and dimensions of each text region in the fronto-parallel view. Then, the filter state is initialized as:

$$\mathbf{x}_0 = \begin{bmatrix} \mathbf{h}_0{}^\mathsf{T} & 0 & 0 & 0 \end{bmatrix}^\mathsf{T}, \tag{2.22}$$

with $\mathbf{h}_0 = \mathcal{H}^{-1}(\mathbf{H}_0)$ and $\mathbf{H}_0 = (\mathbf{H}')^{-1}$ being the initial homography estimation of the transformation between the fronto parallel representation to image coordinates.

On creation, a tracker is marked as unstable. This means that it will not be considered for feature addition, for recognition or transformation to speech, and it will not be shown as a segmented region. It is only considered stable after it is tracked continuously for a number of frames – in our case this was arbitrarily set to ten. As a text region is consistently segmented in a sequence of frames, as opposed to noisy regions, this process cleans most of the text segmentation false positives.

Feature addition When a stable tracker matches some of the measurements inside a text entity, the remaining unmatched measurements are assumed to belong to the same entity. Hence, they are added to the tracker as new features. The corner points of the created feature are mapped to the tracker coordinates using the tracker's state homography, and the observation vector length is increased accordingly.

Tracker removal When a tracker has been regarded a bad track for a few frames because none or too few of its features are matched, the tracker is removed. There is no long term registry

of old trackers. If a tracker is removed (e.g., because it is no longer in view, or due to a long occlusion), and afterwards the text entity it was tracking is detected again, it will be added again as a new tracker. We find this to be an adequate compromise for efficient and long periods of continuous operation.

Occlusions These mechanisms allow our system to deal with brief occlusions of the tracked text regions. A full occlusion will produce bad tracks for the affected trackers. If the occlusion is shorter than the number of frames needed to delete a tracker, when the text region is in view again it will be recovered. The system will also be able to recover the track even with big translations or wide baseline changes of orientation thanks to the use of high level features. Partial occlusions of text regions will be also dealt with in the same fashion, and even on tracker creation, due to the feature addition mechanism.

2.5. OCR and Speech Synthesis

When a tracker is considered stable, it is then a candidate for recognition. The image quadrilateral enclosing the tracked text entity is rectified to a fronto parallel view using the state homography transformation and then sent to OCR. Recognition is performed in a parallel processing task, so the tracking is maintained in real-time while the recognition runs alongside. The decision on which tracker to recognize is weighted by several factors: whether or not there is already any recognition available for this tracker, the OCR confidence value of previous recognitions, and the elapsed time since the last recognition attempt. A tracker might be sent to OCR for recognition several times, but if the returned confidence value of a new recognition is lower than a previous one, the recognition result with higher confidence is kept.

Speech synthesis is the main user interface of the system, and the main intended communication with the user. Those text regions that have a high enough OCR confidence value and have a stable text tracker are considered for being synthesized into speech. The text is sent to an speech synthesis engine so the recognized text is played back to the user. The candidate texts are queued and prioritised according to the distance to the centre of the image. Regions that stay in view of the camera for long enough might be reproduced several times, but as the region identity is maintained throughout the sequence, this delay can be adjusted for the convenience of the user, i.e. by tracking the text we can avoid the system continuously repeating the same text over and over.

2.6. Text reading prototype

This final section describes our prototype text reading machine and presents an overview of the implementation details. Although the major focus of this work is the exploration of the computer vision techniques to allow real-time scene text understanding on video, the construction of a text reading prototype enabled us to use a platform where these techniques could be tested on real life scenarios.

Our reading machine is composed of a camera mounted on a hat, a pair of headphones also connected to the hat and a processing unit. Placing a camera in a hat is a logical choice as it is

both an unobtrusive location and an ideal position in reference to where the eyes and head point to. Mayol et al. [42] examined possible positions of wearable cameras and concluded that head mounted cameras provide the best possible link with the user's attention. The hardware shown here allows its integration into many varieties of hats, here we have used an ordinary fashion accessory – a *flat-cap*. A picture of the reading hat is shown in Figure 2.16a.

The prototype was designed with emphasis on serviceability and visual appearance. All the electronic components are housed on a plastic plate bent to follow the shape of the flat-cap. This plate is removable, easily allowing the replacement of the hat. To minimize the number of cables, all the devices are connected to a generic USB hub which allows connecting the hat to any USB-enabled computing device, from tablets to fully equipped laptop computers, with a single cable. It is built out of commodity hardware, with a total cost of all the components under 100€: a high definition web camera with adjustable focus (Logitech Quickcam Pro 9000), an USB sound card used for voice feedback to the user through a pair of connected headphones and the USB hub. A view of the inner removable part of the hat with the electronic components is shown in Figure 2.16b.

Regarding the software implementation, as one of the design objectives of our prototype is real-time operation, the system is carefully paralellized to make the best possible use of available processors. We use Intel's Threading Building Blocks (TBB),[1] which implements a task-based parallelization paradigm. It features a high level C++ API for defining parallel constructions, supports nested parallelism and provides automatic scalability. The algorithm steps in Figure 2.1 are implemented as separate stages of a processing pipeline. On multiprocessor machines, TBB is able to schedule different stages on different processors so several frames might be simultaneously processed at any given moment. The system maintains a *global state* and a *transient state*. The transient state is carried forward across the stages of the pipeline. At the end of a frame processing, the transient changes are atomically combined in the global system state. This is easily implemented as the library guarantees strict sequential ordering of the pipeline stages of consecutive frames. OCR is spawned as a task outside the main processing pipeline and thus it is scheduled concurrently with it. This allows to maintain real-time performance for the text tracking while being able to perform longer running processes at the same time and without under- or over-subscribing the available parallelism. We find this to be a superior design in terms of portability and scalability when compared to other approaches (such as e.g., PTAM [31]) in which explicit threads are defined with synchronization mechanisms between them. For OCR we use Tesseract,[2] an open-source OCR engine that provides an adequate API and recognition accuracy.

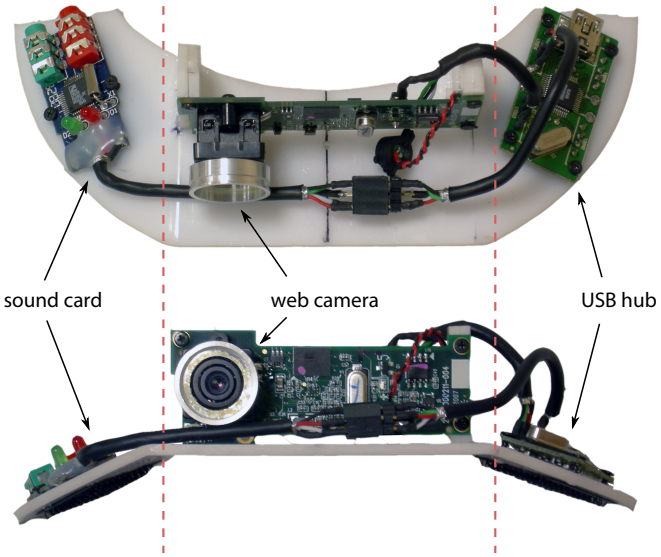
2.7. Conclusion

This chapter has covered the main techniques developed during the course of this PhD: text detection, aggregation, perspective recovery, tracking and other processing stages. In has also presented our prototype wearable text reading machine. Chapter 3 focuses on the evaluation of these techniques and discusses the experimental results.

[1] Threading Building Blocks: `http://threadingbuildingblocks.org/`

[2] Tesseract OCR [73]: `http://code.google.com/p/tesseract-ocr/`

(a)



(b)

Figure 2.16. The reading hat, our prototype text reading machine: (a) the external aspect of the hat and (b) the internal removable components.

Chapter 3

Results and discussion

In this chapter, the results obtained by the proposed system are described and discussed. It is a summary of the results of the works produced over the course of this PhD [43–46, 67], and whose full text is included in Appendices A to E. The chapter is structured as follows. First, Section 3.1 shows the results of the text perspective recovery mechanism. Second, Section 3.2 shows the results of the text tracking mechanism. Finally, Section 3.3 shows performance figures of the operation of the prototype.

3.1. Perspective recovery results

In this section, we evaluate the operation of the perspective recovery subsystem. In our first set of experiments, we used synthetic images to systematically evaluate the performance of the 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. [54]. The nomenclature we use for the axes is illustrated in Figure 3.1: 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 [37] or the Street View Text dataset [81], as neither contains text captured at perspective views; hence they are ill-suited to our purpose here.

3.1.1. 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) [28, 71], 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

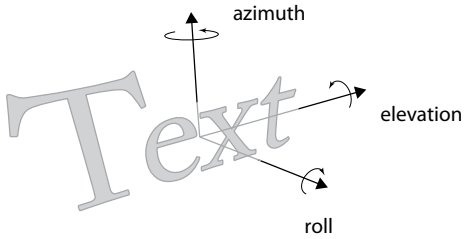


Figure 3.1. Axes for the rotations applied to text in our experiments.

million images; thus each image contains one phrase in a particular orientation. A selection of the images generated is shown in Figure 3.4.

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 [54] 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.

$$\text{accuracy}(\mathrm{R},\mathrm{G}) = 1 - \frac{\min\left(\mathsf{lev}(\mathrm{R},\mathrm{G}),\#\mathrm{G}\right)}{\#\mathrm{G}}\ , \tag{3.1}$$

where R is the text recognized by the method under examination, G is the groundtruth text, $\mathsf{lev}(x,y)$ 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. Figures 3.4i, 3.4m or 3.4p, where the side of the text is blurred).

In Figure 3.2, where the effect of roll is studied, Figure 3.2a shows the performance of the recovery when only in-plane rotations are considered, while Figures 3.2b and 3.2c 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. Figures 3.4l or 3.6h), 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

[1] Tesseract OCR [73]: `http://code.google.com/p/tesseract-ocr/`

keeping the one with higher OCR confidence. As the focus of the evaluation in this section is on the perspective rectification technique, we present the results as is, without this post-processing correction step for this specific and extreme case.

The results in Figure 3.2 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 3.3 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 $45°$ – Figures 3.3a, 3.3c and 3.3e respectively) and likewise, the right column displays the variation of elevation for fixed values of azimuth ($0°$, $30°$ and $45°$ – Figures 3.3b, 3.3d and 3.3f 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 by Myers et al. [54], 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. [54] 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 (Figure 3.3b), the three methods perform similarly. However, when combined with azimuth (Figures 3.3d and 3.3f) 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, Figure 3.5 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 3.1 illustrates the distribution of word lengths in the set of words used in our experiment.

3.1.2. 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

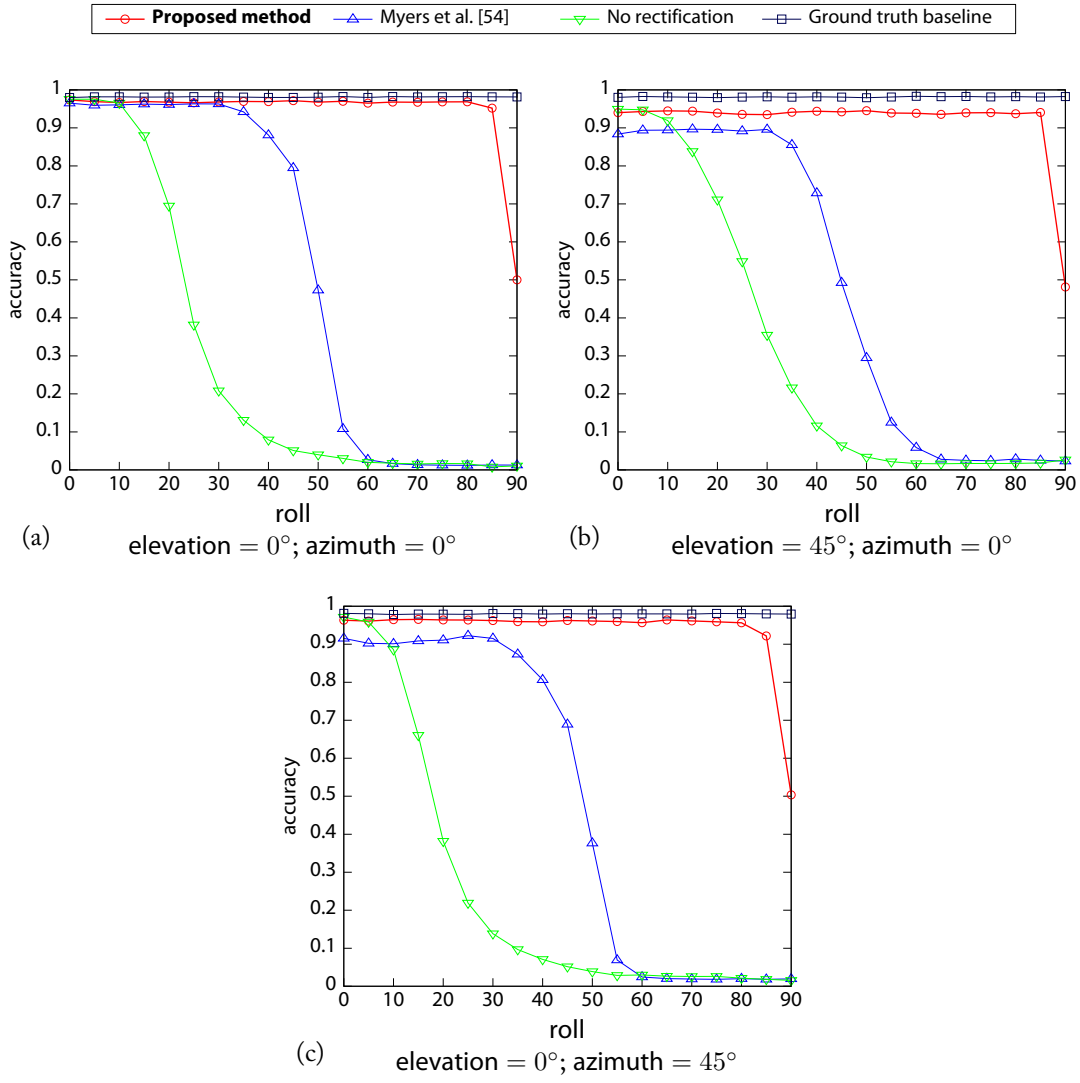


Figure 3.2. The effect of roll on recognition accuracy.

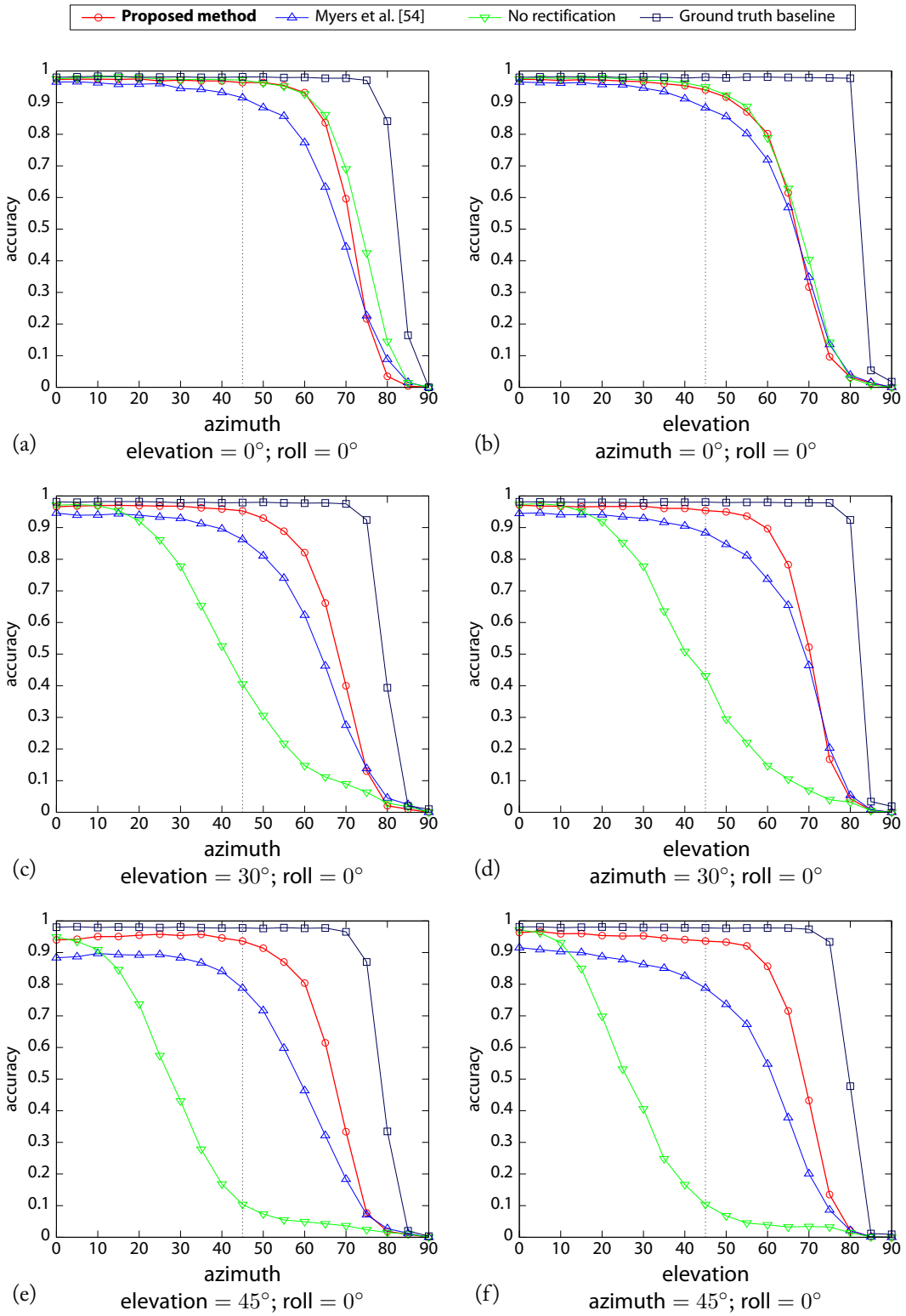


Figure 3.3. The effect of azimuth and elevation on recognition accuracy.

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 3.4. A selection of the synthetic images used in the experiments, along with their *estimated* orientation (red box) and corresponding rectified image.

Figure 3.4. A selection of the synthetic images used in the experiments, along with their *estimated* orientation (red box) and corresponding rectified image. *(cont.)*

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 3.1. Word length distribution in the synthetic text dataset.

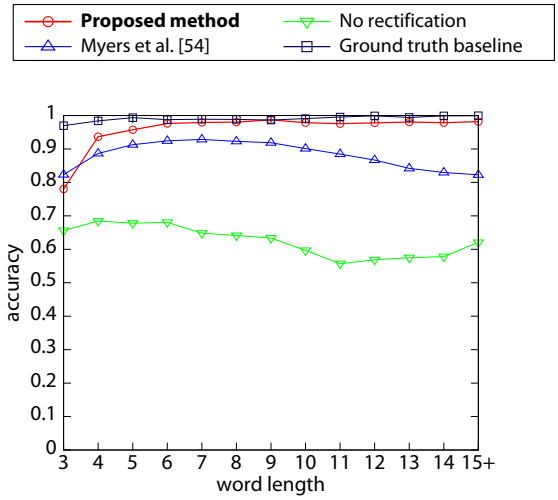


Figure 3.5. The effect of word length on recognition accuracy.

(e.g. serif and sans-serif) and dark and bright colours. Figure 3.6 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 2.1) and corresponding rectified images. The images were manually annotated to obtain a groundtruth of the text present in them. Table 3.2 shows a comparison of the average recognition accuracy, using (3.1), on the unrectified images, and after rectifying with Myers et al.'s [54] method and the proposed method, with the latter showing marked improvement.

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 Figures 3.4g, 3.4i, 3.4o, 3.6a, 3.6d and 3.6f, the 'T' in Figures 3.4a, 3.6f and 3.6g, or the 'W' in Figures 3.4k, 3.6f and 3.6q). 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 Figures 3.6b, 3.6n and 3.6r).

No rectification	0.25
Myers et al. [54]	0.40
Proposed method	0.87

Table 3.2. Average OCR recognition accuracy on the real-world image set.

3.2. Tracking results

In this section, the operation of text tracking mechanism is demonstrated and validated. At first we present a quantitative analysis of the system based on standardized metrics and annotated ground-truth data that help establish a performance baseline for comparative studies. Then, a qualitative evaluation of the prototype's operation in everyday scenarios is outlined to provide an insight into the future improvements and requirements of a text reading system. For our quantitative experiments, three challenging video sequences are used: HOSPITAL, MERCHANT and QUEEN. These contain scene text in a city environment and suffer from hand-held erratic camera motion, as well as blur and a great degree of perspective distortion. The sequences were annotated to obtain a *(i)* ground-truth labelling of text, *(ii)* 3D bounding quadrilaterals, and *(iii)* region identity between frames.[1] To achieve these, tracking was applied in the video sequences using a commercial match-moving software, with each video requiring extensive manual adjustment of the tracked features. After that, 3D editing software was used to locate rectangles in the 3D space around the projected positions of the text in the scene. When the rectangles are projected back as quadrilaterals into the image plane, they perfectly fit the text as seen in the image. The total number of annotated frames is 930. For the qualitative analysis, a variety of example videos, showcasing different scenarios, were experimented with as shown later.

3.2.1. Quantitative analysis of the tracking mechanism

Two distinct tests were performed to evaluate the two desired characteristics of a text tracking system: *(i)* increase in segmentation accuracy and *(ii)* the ability to maintain region identity. The first test is a frame-by-frame comparison of the text detection accuracy between the text segmentation stage alone against the segmented text regions while tracking by our method. The second test evaluates the tracking performance by measuring the detection accuracy along with the region identity across the sequence as a whole.

Frame by frame evaluation

For our text segmentation evaluation, we use the precision and recall measures as introduced in the ICDAR 2003 Robust Reading Competition [37], slightly adapted to operate on arbitrary quadrilaterals instead of rectangles. The degree of match between two quadrilaterals $\mathbf{q}_1$ and $\mathbf{q}_2$

[1] The video sequences and the ground-truth labelling are available at `http://nf.ull.es/q/texttrack`

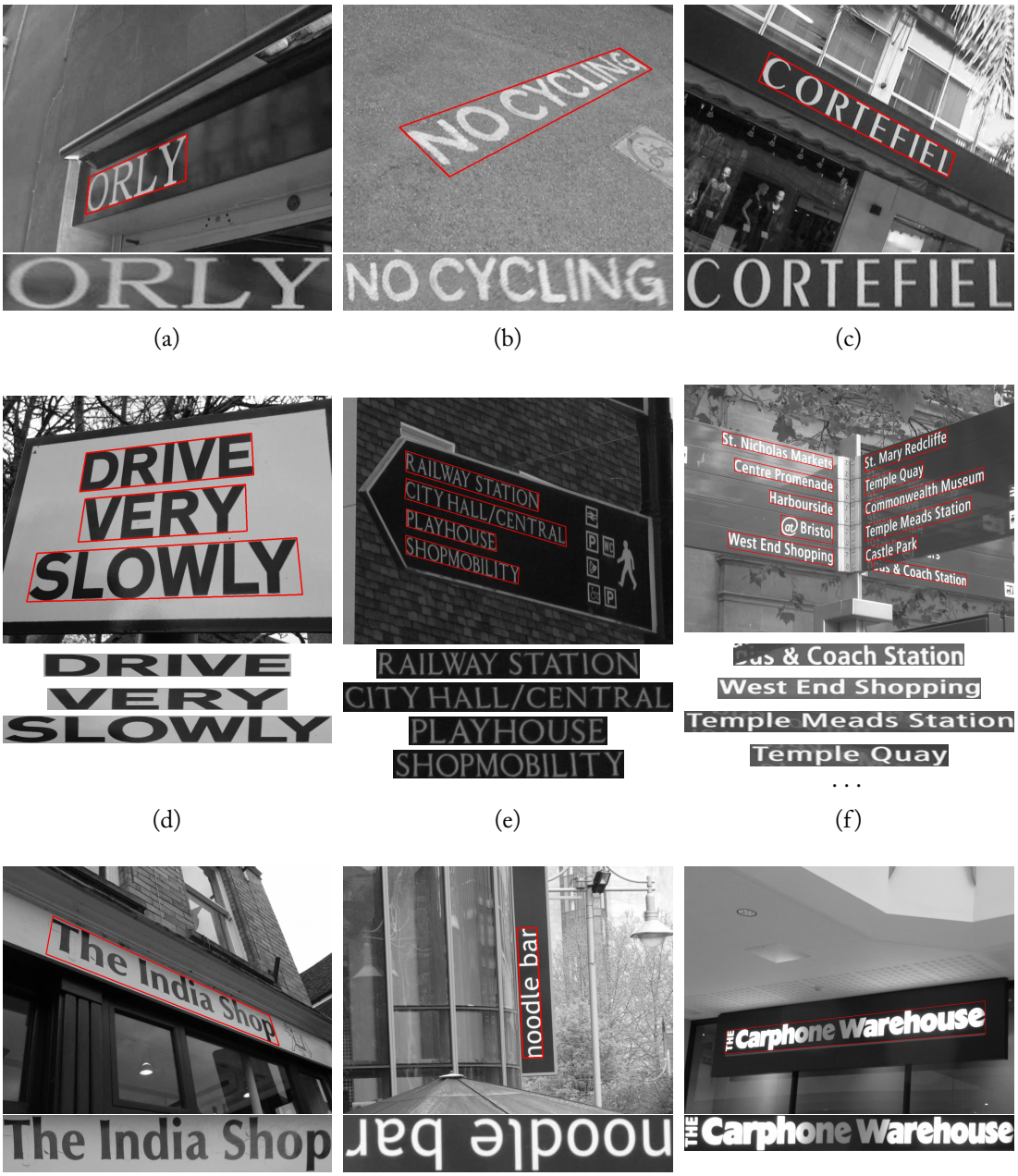


Figure 3.6. A selection of real world images with scene text, along with the text's estimated orientation (red box) and rectified image.

(j) (k) (l)

(m) (n) (o)

(p) (q) (r)

Figure 3.6. A selection of real world images with scene text, along with the text's estimated orientation (red box) and rectified image. *(cont.)*

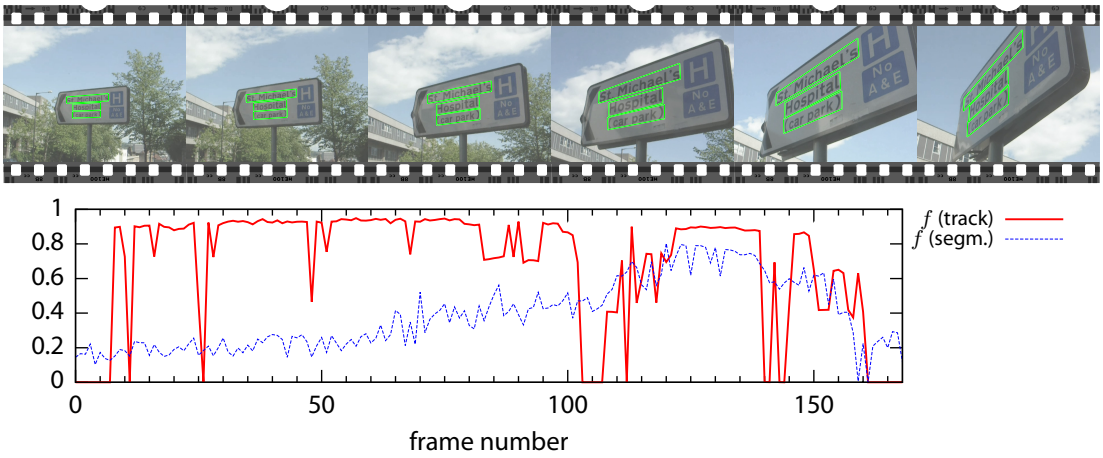


Figure 3.7. Video sequence HOSPITAL. A selection of the frames with tracked text bounding quadrilaterals is shown on top, and the frame by frame segmentation accuracy is shown on the bottom.

is defined as:

$$\mathsf{match}(\mathbf{q}_1, \mathbf{q}_2) = \frac{\mathsf{area}(\mathbf{q}_1 \cap \mathbf{q}_2)}{\mathsf{area}(\mathbf{q}_1 \cup \mathbf{q}_2)} \,. \tag{3.2}$$

When comparing a quadrilateral $\mathbf{q}$ against a set of quadrilaterals Q, the best match is computed as:

$$\mathsf{bestmatch}(\mathbf{q}, Q) = \max_{\mathbf{q}' \in Q} \mathsf{match}(\mathbf{q}, \mathbf{q}') \,. \tag{3.3}$$

Then, the precision p, recall r and f measures for a certain frame are defined as:

$$p = \frac{\sum_{\mathbf{q} \in E} \mathsf{bestmatch}(\mathbf{q}, G)}{|E|} \,, \tag{3.4}$$

$$r = \frac{\sum_{\mathbf{q} \in G} \mathsf{bestmatch}(\mathbf{q}, E)}{|G|} \,, \tag{3.5}$$

$$f = \frac{2}{1/p + 1/r} \,, \tag{3.6}$$

where G is the set of quadrilaterals in the ground-truth and E is the set of quadrilaterals being evaluated.

These measures were computed for each one of the frames in the sequences, comparing the quadrilaterals produced by the text segmentation and orientation detection stages with the quadrilaterals produced by the tracking stage. In our evaluations, we are only considering text lines with 4 or more characters from the ground-truth, as this is the minimum length at which the perspective estimator works reliably [46]. The results are shown in Figures 3.7 to 3.9, where, for every frame in the sequences, the f measure for the segmentation and tracking outputs are plotted in blue and red respectively.

In the HOSPITAL sequence (Figure 3.7), the camera approaches a road sign with strong perspective distortion being introduced gradually, and featuring a very textured tree background,

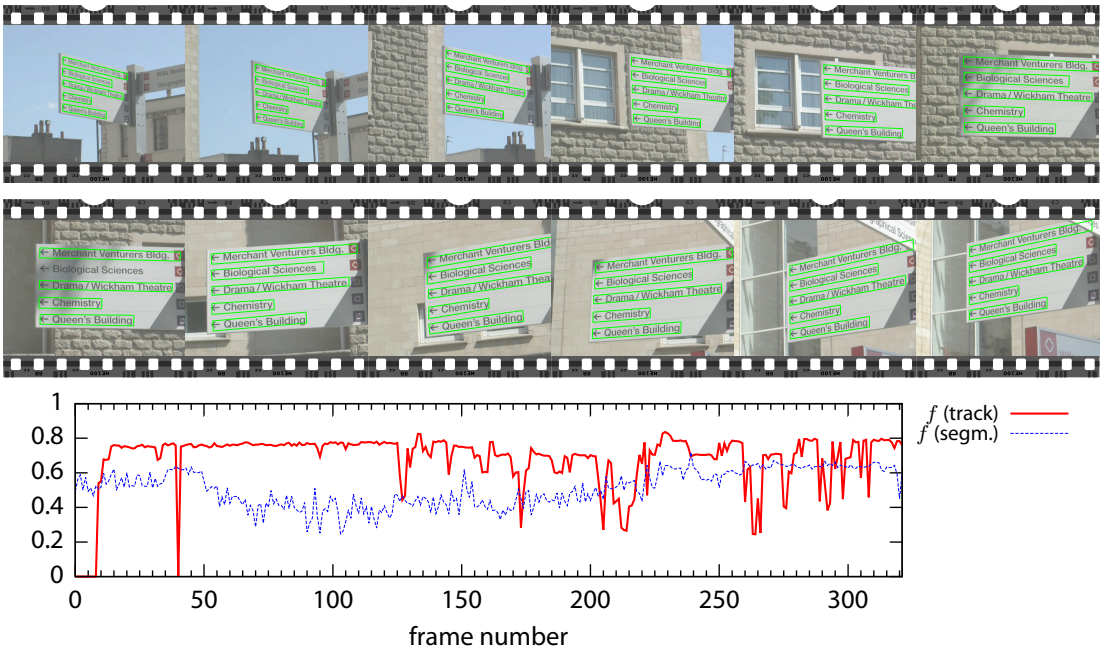


Figure 3.8. Video sequence MERCHANT. A selection of the frames with tracked text bounding quadrilaterals is shown on top, and the frame by frame segmentation accuracy is shown on the bottom.

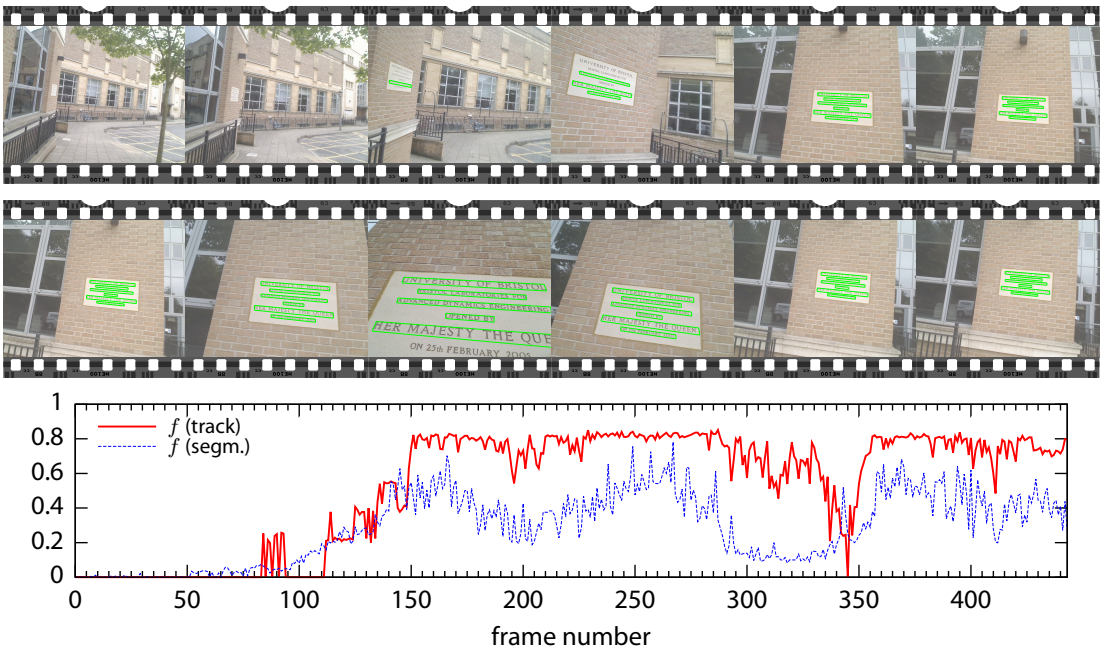
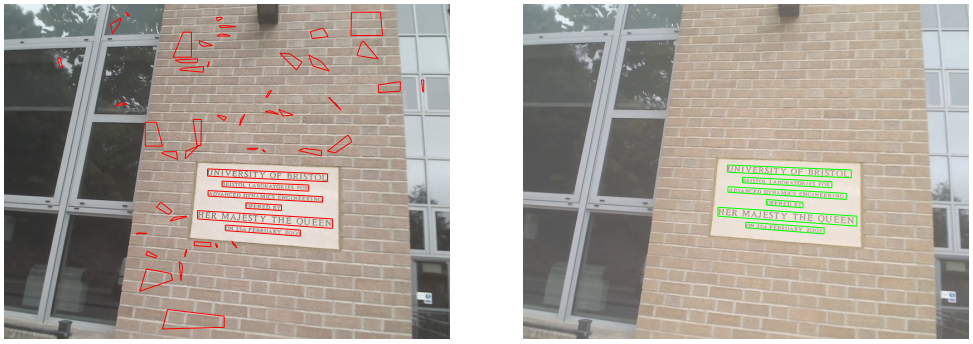


Figure 3.9. Video sequence QUEEN. A selection of the frames with tracked text bounding quadrilaterals is shown on top, and the frame by frame segmentation accuracy is shown on the bottom.



(a) (b)

Figure 3.10. A comparison of the output of the (a) text segmentation and (b) tracking stages, for one frame in the QUEEN sequence.

which makes text segmentation challenging, as can be seen in the performance of the text segmenter. The accuracy of the tracked regions is very good throughout the sequence even when the perspective distortion is significant. The continuous change in perspective occasionally causes the trackers to lose track as the filters converge into the new orientation. This can be seen in the dips around frames 5, 20, and especially between frames 100–115 and at the end of the sequence. Nevertheless, the region identity is never lost, and the trackers recover the lock on the text shortly afterwards.

In the MERCHANT sequence (Figure 3.8), a street sign is panned laterally, with a wide baseline change of orientation and a textured regular brick pattern in the background. Text region tracking is accurate across most of the sequence, as shown in the results. Extreme camera shake is the cause for some of the individual trackers to momentarily lose track, from frame 125 to the end. The system does not produce a box for a text region that is not tracked with enough confidence, thus the alternating and temporary disappearances of some of the boxes across the sequence. Those are promptly recovered without losing the region identity. This is also the reason for the dip in the graph around frame 40, where all the trackers are lost for just one frame.

Finally, the QUEEN sequence (Figure 3.9) features a walkabout towards a building sign which is not visible from the start of the sequence, finishing with a close approach into the text. This very challenging sequence also shows regular window and railing patterns. As can be seen in the results, during the first part of the sequence – until frame 120 – the accuracy is 0 as there is no text. When the text is completely visible, the tracking quickly locks on the text lines and this is clearly shown in the graph between frames 120 and 280. Then, during a camera movement towards the text, there are two brief camera blur events between frames 280–300 and 330–350 that cause the segmentation to produce very few regions, and thus making the tracker to momentarily lose track as there are no regions to track, especially at frame 345. The tracker promptly recovers the track after these events. The ability to quickly recover from failures demonstrates the versatility of the proposed method. To illustrate the effect that text tracking has on segmentation accuracy, Figure 3.10 shows the output of the text segmentation stage, with a considerable number of false positives, compared to the output of the tracking stage, where the spurious regions have been discarded.

Tracking evaluation

The characteristics of our prototype system preclude making a comparative study against any of the few other published text tracking systems. For example, in the work in [55], the text regions are manually selected prior to tracking or in the work in [30] text tracking was performed on overlaid text. Furthermore, we are tracking 3D text quadrilaterals, as opposed to 2D rectangles, which is a fundamental difference with any other published work (e.g., [22, 48]). With this paper, we are also publishing our scene text tracking dataset and its associated ground-truth data – something that has not been done before – in the hope that it will be useful to other researchers and to enable future comparative studies.

However, we do present comparative results against our own previous work in [43]. To evaluate the performance of the text tracking we adopt the CLEAR object tracking metrics as suggested by Bernardin and Stiefelhagen [2] and Kasturi et al. [30]. The metrics are designed to evaluate the detection and tracking performance across a sequence as a whole, and thus penalize false positives and false negatives as well as region identity changes or losses. For every frame k, there is a mapping M_k between the elements in the ground-truth and detected sets:

$$M_k = \{(\mathbf{g}, \mathbf{e}), \text{ with } \mathbf{g} \in G_k \text{ and } \mathbf{e} \in E_k\} , \tag{3.7}$$

where G_k and E_k are the sets of ground-truth and detected quadrilaterals at frame k. Mappings are unique for each element on each set. Our criteria for considering a candidate match between two quadrilaterals is that they have some overlap (e.g. $\mathsf{area}(\mathbf{q} \cap \mathbf{q}') > 0$). To select a unique match between the candidates, we first consider the same match as in the previous frame if they still overlap; otherwise the pair with maximum overlap is chosen (refer to [2] for the full explanation and rationale of the matching strategy). Every detected quadrilateral that is not mapped is a false positive; likewise, every unmapped ground-truth quadrilateral is a missed detection. If a quadrilateral $\mathbf{g} \in G_k$ is matched to different quadrilaterals $\mathbf{q}, \mathbf{q}' \in E_k$ in consecutive frames, it is considered an identity mismatch.

Two measures are defined, the *multiple object tracking precision* (MOTP) and the *multiple object tracking accuracy* (MOTA):

$$\mathsf{MOTP} = \frac{\sum_k \sum_{(\mathbf{g},\mathbf{e}) \in M_k} \mathsf{match}(\mathbf{g}, \mathbf{e})}{\sum_k |M_k|} , \tag{3.8}$$

$$\mathsf{MOTA} = 1 - \frac{\sum_k (\delta_k + \phi_k + \log_{10}(\rho_k))}{\sum_k |G_k|} , \tag{3.9}$$

where δ_k, ϕ_k and ρ_k are the total number of missed detections, false positives and id mismatches for frame k respectively.

Table 3.3 summarizes the results of our proposed method in comparison to our previous text tracking technique using Particle Filters [43]. Our PF method had a simple 2D state model and did not perform any perspective estimation and this is shown in the MOTP values, where the proposed method consistently achieves very high values (0.89, 0.80 and 0.78 for the HOSPITAL, QUEEN and MERCHANT sequences respectively) thanks to the enhanced state model. It also produces high MOTA values (0.82, 0.70 and 0.59 for the MERCHANT, HOSPITAL and QUEEN

	Proposed method		PF method [43]	
Sequence	MOTP	MOTA	MOTP	MOTA
HOSPITAL	0.89	0.70	0.10	0.32
MERCHANT	0.78	0.82	0.16	0.24
QUEEN	0.80	0.65	0.10	0.13

Table 3.3. Whole sequence tracking evaluation.

sequences) due to the low number of false positives and id mismatches that the tracking produces. As previously explained, the challenging sequences used in our experiments feature very erratic camera motion, blur, and perspective distortion. Our MOTA evaluation is penalized on those frames where the text is not tracked with enough confidence, in which our system does not produce any box, getting counted as a missed detection.

3.2.2. Qualitative evaluation

We show more examples in Figure 3.11 to further illustrate the operation of the proposed method. In the CLIFTON sequence (Figure 3.11a), the text is never in a fronto-parallel, horizontal orientation with respect to the camera and undergoes a wide baseline change in orientation. This is also true for the WOLFGANG sequence (Figure 3.11c), which also features a complex background and very blurred frames due to camera vibrations. The HANNOVER sequence (Figure 3.11b) contains regular, very contrasted tiles in the background which produce a great number of false positives from the text segmentation stage. The BYRON PLACE sequence (Figure 3.11d) features a change in orientation around elevation. Note, the text is not visible at the start of the sequence, but as soon as it appears, the system is able to pick up the location of the various lines of text and then track them continuously. As with previous sequences (i.e., MERCHANT and QUEEN), camera shake is responsible for the momentary disappearance of tracked regions, after which the tracker recovers without region identity loss. Finally, the UOB sequence (Figure 3.11e) features a moving partial occlusion across the tracked text. As there are always enough visible features (i.e., characters) for each one of the text lines, the proposed method is able to keep the identity and location of all the text regions in the scene. For all the sequences, the system is able to quickly spot the text in its original orientation and maintain the region identity throughout the duration of the video.

3.3. Performance results

In this final section some performance measures about the proposed system are discussed. The experiments were run on a standard PC with an Intel Core 2 Quad Q6600 processor and 8Gb of RAM. The system operates at video rate (average 25fps) on the video sequences used for our experiments. A breakdown of the times spent by the algorithm on each of the stages is presented in Table 3.4. The most expensive stage in terms of computation time is the tracking observation,

(a) Sequence CLIFTON

(b) Sequence HANNOVER

(c) Sequence WOLFGANG

(d) Sequence BYRON PLACE

(e) Sequence UOB

Figure 3.11. Video sequences from the qualitative evaluation experiment.

acquisition	5ms
segmentation	22ms
aggregation	5ms
perspective estimation	27ms
tracking: prediction	8ms
tracking: observation	42ms
OCR	250ms

Table 3.4. Time spent on each stage of the algorithm.

which includes the feature matching. The text segmentation and perspective estimation are also major contributors to the time needed to process one frame, although, as a reference, perspective estimation requires on average 0.1 ms per text line. Those stages are split into a pipeline and are automatically distributed between the processor cores to achieve parallelism. The OCR task is comparatively much slower than the rest of the stages together. This demonstrates one of the benefits of text tracking: OCR runs on an independent thread and thus does not contest with the main processing pipeline to achieve the desired frame rate. When the recognition results are ready, the region identity maintained by the text tracking is used to assign the recognized text to the correct text region.

3.4. Conclusion

In this chapter, the results obtained during this PhD have been presented and discussed. We have shown that our method for text perspective recovery improves the previous state-of-the-art in terms of ranges of angles and speed. We have also demonstrated the operation of our end-to-end tracking system in outdoor scenarios. Finally, Chapter 4 concludes this dissertation and provides the summary of the contributions of this work.

Chapter 4

Conclusions

This chapter concludes the dissertation and gives directions for future research. An itemised summary of the contributions of this work is presented in Section 4.1.

We have presented a text reading system based on text tracking. Aimed at scene text, it focuses on isolated words or short sentences, as found on billboards, posters, shop names, street signs, etc. It operates autonomously and in real-time, automatically detecting and recognising new text regions and discarding the old ones. The end-to-end text tracking system has required the development of novel techniques for fast text detection, text aggregation, and text perspective recovery. We have shown quantitative and qualitative analyses of the performance and capabilities of our prototype including a detailed analysis of the perspective recovery stage which significantly improves the previous state-of-the-art. We also released our scene text tracking dataset and its associated ground-truth data to enable future comparative studies. Additionally, although it has not been the primary focus of this PhD, we have also constructed a wearable text reading prototype which can be used as a platform to showcase the operation of the developed system.

The area of text tracking is very young and there is still a lot to be accomplished. We think we have presented a novel step towards fast text segmentation and perspective aware text tracking. However, a number of shortcomings and newer goals are yet to be addressed. Our method is focussed on larger text and is not suited to deal with smaller document texts. Also, as a matter of trading accuracy for speed, we do not necessarily use state-of-the-art text segmentation. There is also scope to improve 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.

There are other avenues which could be explored. For example, the OCR results from multiple frames could be combined to obtain a more accurate global recognition (e.g. to bypass reflections and occlusions). Rectified images from several frames could be integrated to help construct super-resolution and/or larger (by mosaicing) images. Moreover, we still have to study and understand how the tracking information can help build a better user interface for assistive devices. Observing the patterns of movement and context in the surroundings is crucial for deciding when and how to read text back to the user, enabling a more useful interaction experience. In order to identify the aspects of the user interaction that can benefit from the framework presented here, it is vital to conduct studies with people who have visual impairments.

Further research is needed to explore and evolve the technologies presented here.

4.1. Summary of contributions

Main contributions:

- A novel method for efficient scene text perspective recovery has been developed. 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.
- A novel method for perspective aware scene text tracking has been developed. We have shown quantitative and qualitative analysis of the performance and capabilities of our prototype.
- We have released our scene text tracking dataset and its associated ground-truth data to enable future comparative studies.

Additional contributions:

- A novel method for scene text detection based on the hierarchical filtering of Adaptive Thresholding (AT) has been developed.
- A novel method for scene text detection based on the hierarchical filtering of Maximally Stable Extremal Regions (MSERs) has been developed.
- A novel method for multi orientation text aggregation based on Delaunay graphs has been developed.

Appendix A

A Framework Towards Realtime Detection and Tracking of Text

This appendix includes the full text for the following article:

Title	A Framework Towards Realtime Detection and Tracking of Text.
Authors	Carlos Merino and Majid Mirmehdi
Type	Conference proceedings
Conference	Camera-Based Document Analysis and Recognition
Year	2007
Pages	10–17
URL	`http://www.imlab.jp/cbdar2007/proceedings/papers/O1-2.pdf`

A Framework Towards Realtime Detection and Tracking of Text

Carlos Merino
Departamento de Fisiología
Universidad de La Laguna
38071 Santa Cruz de Tenerife, Spain
cmerino@ull.es

Majid Mirmehdi
Department of Computer Science
University of Bristol
Bristol BS8 1UB, England
majid@cs.bris.ac.uk

Abstract

We present a near realtime text tracking system capable of detecting and tracking text on outdoor shop signs or indoor notices, at rates of up to 15 frames per second (for generous 640×480 *images), depending on scene complexity. The method is based on extracting text regions using a novel tree-based connected component filtering approach, combined with the Eigen-Transform texture descriptor. The method can efficiently handle dark and light text on light and dark backgrounds. Particle filter tracking is then used to follow the text, including SIFT matching to maintain region identity in the face of multiple regions of interest, fast displacements, and erratic motions.*

1. Introduction

Tracking text is an important step towards the identification and recognition of text for outdoor and indoor wearable or handheld camera applications. In such scenarios, as the text is tracked, it can be sent to OCR or to a text-to-speech engine for recognition and transition into digital form. This is beneficial in many application areas, such as an aid to the visually impaired or for language translation for tourists. Furthermore, the ability to automatically detect and track text in realtime is of use in localisation and mapping for human and robot navigation and guidance.

A review [9] and some collections of recent works [2, 1] in camera-based document analysis and recognition, highlight substantial progress in both single image and multiframe based text analysis. Overall, there have been relatively few works on general text tracking. Multiframe text analysis has been mainly concerned with improving the text in a super-resolution sense [12] or for continuous recognition of text within a stationary scene e.g. on whiteboards or in slideshows [18, 20].

A directly related work in the area of general scene text tracking is by Myers and Burns [13] which successfully tracks scene text undergoing scale changes and 3D motion. However, this work applies to tracking in batch form and is not a realtime solution. Also in [13], the text detection is done by hand, manually indicating a starting bounding box for the tracking system to follow. Another work of interest is Li *et al.*[8] in which a translational motion tracking model was presented for caption text, based on correlation of image blocks and contour based stabilisation to refine the matched position. Less directly related, in [16], seven simple specific text strings were looked for by a roving camera from a collection of 55 images in an application to read door signs.

The focus of this paper is on the development of a resilient text tracking framework, using a handheld or wearable camera, as a precursor for our future work on text recognition. The only assumption we make is that we are looking for larger text sizes on shop and street signs, or indoor office boards or desktop documents, or other similar surfaces. Our proposed method is composed of two main stages: candidate text region detection and text region tracking. In the first stage, regions of text are located using a connected components approach combined with a *texture* measure step [17] which to the best of our knowledge has never been applied to text detection; this provides candidate regions or components which are then grouped to form possible words. The method is highly accurate but not infallible to noise, however, noisy or non-text candidate regions are not detected as persistently as true text regions, and can be rejected forthright during the tracking step. In the second stage, particle filtering is applied to track the text frame by frame. Each hypothesised system state is represented by a particle. The particles are weighted to represent the degree of belief on the particle representing the actual state. This non-linear filtering approach allows very robust tracking in the face of camera instability and even vigorous shakes. SIFT matching is used to identify regions from one frame to the next. We describe the details of this framework in the next few sections.

2. Background

It should be noted that there is a significant body of work on detecting (graphical) text that has been superimposed in images and videos, as well as in tracking such text. Example works are [10, 8]. In this work we concentrate solely on text embedded in natural scenes.

Segmentation of text regions involves the detection of text and then its extraction given the viewpoint. For example, almost each one of the works published in [2, 1] present one method or another for text segmentation, usually from a fronto-parallel point of view. Example past works considering other viewpoints and recovering the projective views of the text are [4, 14, 13]. Although in this work we engage in both *segmenting and tracking* text involving varying viewpoints, actual fronto-parallel recovery is not attempted. This is a natural step possible from the tracking motion information available and will be a key focus of our future work.

An issue of note is the problem of scale. Myers and Burns [13] dealt with this by computing homographies of planar regions that contain text, and when computationally tractable, this could be useful for any (realtime) text tracking application. Here, we are detecting text dynamically, hence at some smaller scales our detector will simply not find it, until upon approach it becomes large enough.

3. Methodology

The text tracking framework proposed here is based around the principle of a *tracker* representing a *text entity* - a word or group of words that appear together in an image as a salient feature, where each word comprises two or more components or regions. Trackers are dynamically created when a new text entity is detected; they follow the text frame to frame, and they get removed when the text cannot be detected anymore. Partial occlusion is dealt with, and in cases of full occlusion, a new tracker starts once the text is back in view. Our text tracking framework involves text segmentation, text region grouping, and tracking, including dynamic creation and removal of trackers.

3.1. Text segmentation

The text segmentation stage uses a combination of a connected components (CC) approach and a region filtering stage, with the latter involving the novel application to text analysis of a *texture* measure. The resulting component regions are then grouped into text entities.

3.1.1 Connected component labelling Following CC labelling in [7], León *et al* employed a tree pruning approach to detect text regions. They thresholded the image at every grey level, and built a Max-tree representation where each node stored the CC of the corresponding threshold level.

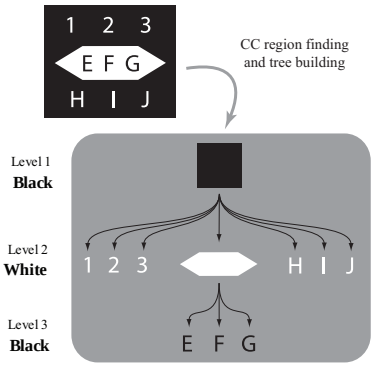


Figure 1. A synthetic sample image and its corresponding tree of connected regions.

The leaves of the tree represented the zones whose grey levels were the highest in the image. For detection of dark text over bright backgrounds, they built a different tree, a Min-tree, where the leaves represented the zones with the lowest grey levels in the image. This two pass treatment of bright text and dark text is very common in text detection algorithms.

We improve on the tree region labelling method in [7] by introducing a simple representation that allows efficient, one pass detection of bright text (white over black) and dark text (black over white) in the same tree. Initially, simple local adaptive thresholding is applied to the source frame. We empirically fixed the local threshold window size to 17×17 throughout all our experiments. The threshold was the mean grey level value of the window itself. Connected component region labelling is then performed on the thresholded image. This labelling builds a tree of connected regions, with the outermost region the root of the tree and the innermost regions the leaves. We allow the regions to toggle their label value from black to white as we go down each level of the tree. The tree represents the nesting relationship between these regions. Each node of the tree keeps only the *contour* around the border of the regions (see Figure 1).

Once the tree is built, it is walked depth-first with the objective to filter out the regions that are not text. Each node of the tree is classified as text or non-text during the walk using region filtering as described later below.

Usually, on real-world images with scene text, structural elements (such as sign borders, posters frames, etc.) can exhibit characteristics of text, such as high contrast against their backgrounds or strong texture response. These elements can be easily discarded (as long as they are not at a leaf) using the nesting relationship present in the proposed tree. When a node has children already classified as text,

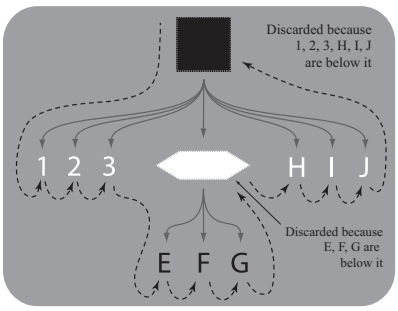


Figure 2. Parent nodes are discarded when children are classified as text.

it is discarded as non-text, despite the text classifying functions may having marked it as text. This discards most of the non-text structural elements of the text (Figure 2).

3.1.2 Region filtering To classify text regions we apply three tests in cascade, meaning that if a test discards a region as non-text, no more tests are applied to it. This is in a similar fashion to Zhu *et al.* [21] who used 12 classifiers. In our case, the fewer tests are important for real time processing, and coarse, but computationally more efficient tests are applied first, quickly discarding obvious non-text regions, and slower, more discriminative tests are applied last, where the number of remaining regions is fewer. The test we apply are on *size*, *border energy*, and an eigenvector based *texture measure*.

Size - Regions too big or too small are discarded. The thresholds here are set to the very extreme. Very small regions are discarded to avoid noise. This may still drop characters, but they probably would be otherwise impossible to recognise by OCR and as the user gets closer, they are more likely to be picked up anyway. Large regions are discarded because it is unlikely that a single character occupies very large areas (over half the size) of the image.

Border energy - A Sobel edge operator is applied to all the points along the contour of each component region r and the mean value is obtained:

$$B_r = \frac{\sum_{i=1}^{P_r} \sqrt{(G_{ix}^2 + G_{iy}^2)}}{P_r} \qquad (1)$$

where P_r denotes the number of border pixels in region r, and G_x and G_y represent the Sobel gradient magnitudes. This is referred to as the *border energy* and provides a measurement of region to background contrast. Regions with border energy value below a very conservatively fixed threshold are discarded. This removes regions that usually appear in less textured and more homogeneous regions.

Jiang *et al* [6] used a three level Niblack threshold [19] in their text detection technique with good results. This introduces the local pixel values variance into the threshold calculation. However, this involves computing the standard deviation of local pixel values and we have found that doing a simpler adaptive threshold and afterwards discarding the noisy regions is faster. Also, the proposed tree walking algorithm transparently manages bright-text and dark-text occurrences on the same image without the need to apply a three level threshold image.

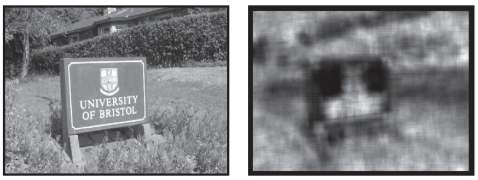


Figure 3. Original image and its Eigen-Transform response.

Texture measure - For this final decision-making step we apply a texture filter whose response at positions within the region pixels and their neighbourhoods is of interest.

We have previously combined several texture measures to determine candidate text regions, see [3]. These were mainly tuned for small scale groupings of text in the form of paragraphs. Although quite robust, the need for faster processing precludes their combined use. Here, we introduce the use of the Eigen-Transform texture operator [17] for use in text detection. It is a descriptor which gives an indication of surface roughness. For a square $w \times w$ matrix representing a pixel and its neighbouring grey values, the eigenvalues of this matrix are computed: $\|\lambda_1\| \geq \|\lambda_2\| \geq \ldots \|\lambda_w\|$. The largest l eigenvalues are discarded since they encode the lower frequency information of the texture. Then, the Eigen-Transform of the central pixel is the mean value of the $w - l + 1$ smaller magnitude eigenvalues:

$$\Gamma(l, w) = \frac{1}{w - l + 1} \sum_{k=l}^{w} \|\lambda_k\| \qquad (2)$$

The Eigen-Transform has a very good response to texture which exhibit high frequency changes, and we found in our experiments that it responds to text very well for this reason, see a simple example in Figure 3 where both the text and the background texture are picked up well. It can, however, be a fairly slow operator, but fortunately we need only apply it to the component region pixels. Indeed, we compute the Eigen-Transform only on some regularly sampled points inside the bounding box of each region of interest. A key factor in (2) is the size of w. This is determined automatically by setting it dynamically according to the height

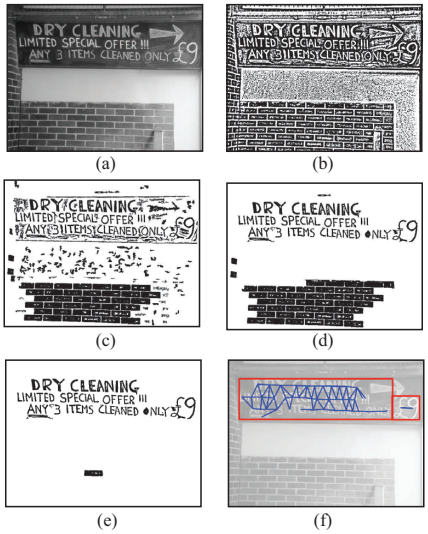


Figure 4. Steps of the text segmentation and grouping. (a) Original image, (b) Adaptive threshold, (c)–(e) result after filtering by size, border energy and Eigen-transform measure, (f) perceptual grouping.

of the region under consideration. Then l is set to be a fraction of w.

The result of the text segmentation stage is a set of candidate regions with a high likelihood of being text. For each region, the centre position of its bounding box is stored as a component $\mathbf{c}_i$ into the *observation function* $\mathbf{y}_k$ of the particle filter (see section 3.2). As a result of the CC region tree design, and taking into account only the contour and not the contents, both inverted text (light on dark) and normal text (dark on light) are detected in the same depth-first pass. Figure 4 shows an example result highlighting each of the text segmentation and grouping steps.

3.1.3 Perceptual text grouping - The text grouping stage takes the regions produced by the text segmentation step and makes compact groups of perceptually close or *salient* regions. We follow the work by Pilu [14] on perceptual organization of text lines for deskewing. Briefly, Pilu defines two scale-invariant saliency measures between two candidate text regions A and B: *Relative Minimum Distance* λ and *Blob Dimension Ratio* γ:

$$\lambda(A,B) = \frac{D_{\min}}{A_{\min} + B_{\min}} \qquad \gamma(A,B) = \frac{A_{\min} + A_{\max}}{B_{\min} + B_{\max}} \quad (3)$$

where $D_{\min}$ is the minimum distance between the two regions, and $A_{\min}$, $B_{\min}$, $A_{\max}$ and $B_{\max}$ are respectively the minimum and maximum axes of the regions A and B. Pilu's text saliency operator between two text regions is then:

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

where $N(x,\mu,\sigma)$ is a Gaussian distribution with mean μ and standard deviation σ whose parameters were determined experimentally in [14]. To reduce the complexity of comparing all the regions against each other, we construct a planar graph using Delaunay triangulation, with the region centres as vertices. The saliency operator is then applied to each edge of this graph, keeping only the salient ones and removing the rest. This edge pruning on the graph effectively divides the original graph into a set of connected subgraphs. Each subgraph with more than two vertices is considered a text group. This additional filtering step removes a number of isolated regions (see Figure 4(f)).

3.2. Text tracking

Particle filtering, also known as the Sequential Monte Carlo Method, is a non-linear filtering technique that recursively estimates a system's state based on the available observation. In an optimal Bayesian context, this means estimating the *likelihood* of a system's state given the observation $p(\mathbf{x}_k|\mathbf{y}_k)$, where $\mathbf{x}_k$ is the system's state at frame k and $\mathbf{y}_k = \{\mathbf{c}_1,\ldots,\mathbf{c}_K\}$ is the observation function.

Each hypothesised new system state at frame k is represented by a particle resulting in $\{\mathbf{x}_k^{(1)},\mathbf{x}_k^{(2)},\ldots,\mathbf{x}_k^{(N)}\}$, where N is the number of particles. Each particle $\mathbf{x}_k^{(n)}$ has an associated weight $\left\{(\mathbf{x}_k^{(1)},w_k^{(1)}),\ldots,(\mathbf{x}_k^{(N)},w_k^{(N)})\right\}$ where $\sum_{i=1}^{s} w_k^{(i)} = 1$. Given the particle hypothesis $\mathbf{x}_k^{(n)}$, the weights are proportional to the likelihood of the observation, $p(\mathbf{y}_k|\mathbf{x}_k^{(n)})$. For a detailed explanation of particle filter algorithms and applications, see e.g. [5].

Particle filtering is the ideal method given the instability of the handheld or wearable camera in our application domain. We build on the particle tracking framework developed in [15] for simultaneous localisation and mapping (SLAM). Here we want to independently track multiple instances of text in the image, with a simple state representation. Thus, each text entity is assigned a particle filter, i.e. a *tracker*, responsible of keeping its state. The main components to now deal with in a particle filter implementation are the *state representation*, the *dynamics model* and the *observation model*.

3.2.1 State representation - The *tracker* represents the evolution over time of a text entity. It has a state that tries to model the apparent possible changes that the text entity may experience in the image context. The model has to be rich enough to approximate the possible transformations of the text but at the same time simple enough to be possible to estimate it in real time.

The state of a tracker at frame k is represented by a 2D translation and rotation: $\mathbf{x}_k = (t_x, t_y, \alpha)$. We found this simple state model provides sufficient accuracy given the degree of movement within consecutive frames, but is also important in computational savings towards a real-time model[1]. This state defines a relative coordinate space, where the x-axis is rotated by an angle α with respect to the image, and its origin is at (t_x, t_y) in image coordinates.

Let's say a text entity contains M components. Its tracker preserves a list of M features $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_M\}$ where each feature $\mathbf{z}_i$ is a 2D position lying in the tracker's relative coordinate space. Each feature represents a text component being tracked, and it is fixed during tracker initialization. We define the transformation function $\Psi(\mathbf{z}_i, \mathbf{x}_k)$ as the coordinate transform (translation and rotation) of a feature position from the state's coordinate space to image coordinates. This is used during weighting. Additionally, each feature is associated with a set of SIFT descriptors [11] computed only once during the tracker initialization. They give the ability to differentiate between candidate text components, providing a degree of multiscale and rotation invariance to the feature matching as well as resilience to noise and change in lighting conditions[2].

Figure 5 shows the current state representation $\mathbf{x}_k$ of a tracker at frame k which has $M = 4$ features $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4\}$. For ease of exposition, all the features are visualised to lie along the x-axis of the tracker's coordinate space. Further, the figure shows another particle $\mathbf{x}_k^{(1)}$ representing an alternative state hypothesis. The four features $\mathbf{z}_i \in \mathbf{Z}$ are mapped to the particle's relative coordinate space to show the same set of features from a different reference frame. The observation function $\mathbf{y}_k$, with $\mathbf{y}_k = \{\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4\}$ representing the center points of the candidate text components is also shown.

3.2.2 Dynamics model - The dynamics model defines the likelihood of a system state transition between time steps as $p(\mathbf{x}_k|\mathbf{x}_{k-1})$. It is composed of a deterministic part - a prediction of how the system will evolve in time, and a stochastic part - the random sampling of the particles around the predicted position. Examples of prediction schemes are constant position, constant velocity and constant acceleration. Examples of stochastic functions are uniform and Gaussian random walks around an uncertainty window of the predicted position.

The selection of an appropriate dynamics model is crucial for a tracking system to be able to survive unpredictable movements, such as those caused by wearable or hand-held camera movements. Pupilli [15] concluded that for such scenarios a constant position prediction model with a uniform or Gaussian random walk would provide better results, due to the unpredictable nature of erratic movements. Here, we follow this advice to use a constant position model with random Gaussian walk around the last state, i.e. $p(\mathbf{x}_k|\mathbf{x}_{k-1}) = N(\mathbf{x}_{k-1}, \Sigma)$. The covariance matrix Σ defines the *particle spread* which is empirically set to a generous size, and automatically reduced via an annealling process as in [15].

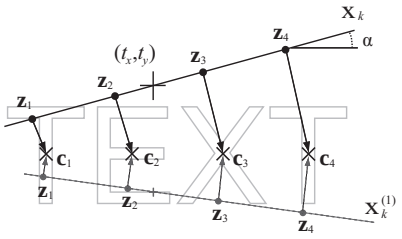


Figure 5. State model of one *tracker*, $\mathbf{x}_k = (t_x, t_y, \alpha)$, with 4 tracked features $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4\}$. A particle, $\mathbf{x}_k^{(1)}$, shows a different state hypothesis.

3.2.3 Observation model - Given a particle state hypothesis, the observation model defines the likelihood of the observation, $p(\mathbf{y}_k|\mathbf{x}_k^{(n)})$. The weight of each particle is calculated based on the comparison from projected features' positions and actual text components found in the image. An inlier/outlier likelihood proposed by Pupilli [15] is used.

For each tracked feature $\mathbf{z}_i \in \mathbf{Z}$, a set of candidate components $\mathbf{y}_{ki} \subseteq \mathbf{y}_k$ $\{(\mathbf{z}_1, \mathbf{y}_{k1}), (\mathbf{z}_2, \mathbf{y}_{k2}), \ldots, (\mathbf{z}_M, \mathbf{y}_{kM})\}$ is computed, based on their matching to the SIFT descriptors previously stored for each feature. This reduces the search space of the particles and gives robustness to the tracking process.

The weight of a particle is proportional to the number of observed candidate components inside a circular region of radius ε around each tracked feature. First an *inlier threshold* function $\tau(\mathbf{a}, \mathbf{b})$ is defined:

$$\tau(\mathbf{a}, \mathbf{b}) = \begin{cases} 1 & \text{if } d(\mathbf{a}, \mathbf{b}) < \varepsilon \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where $d(\mathbf{a}, \mathbf{b})$ is the distance between two points. Then, the likelihood is:

$$p(\mathbf{y}_k|\mathbf{x}_k^{(n)}) \propto \exp\left(\sum_{\mathbf{z}_i \in \mathbf{Z}} \sum_{\mathbf{c}_j \in \mathbf{y}_{ki}} \tau\left(\Psi(\mathbf{z}_i, \mathbf{x}_k^{(n)}), \mathbf{c}_j\right)\right) \quad (6)$$

[1] However, we intend to investigate more complex motion models in future while ensuring the realtime aspects of the work are not compromised

[2] Note to Reviewers: We have found the SIFT matching to grossly slow our system down. By the time of this Workshop we will have implemented and hope to report faster invariant feature matching using e.g. the Hessian Affine or MSER which will additionally give a greater degree of affine invariancy

where $\Psi(\mathbf{z}_i, \mathbf{x}_k^{(n)})$ is the transformation function defined in subsection 3.2.1. Figure 6 shows the weighting process of one feature $\mathbf{z}_2$ for two different hypothesis, $\mathbf{x}_k^{(1)}$ and $\mathbf{x}_k^{(2)}$. The latter is nearer to the actual state of the system and gets a greater weight. Note that for illustration purposes we are considering here that the candidate group components for feature $\mathbf{z}_2$ is all the observation: $\mathbf{y}_{k2} = \mathbf{y}_k = \{\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4\}$.

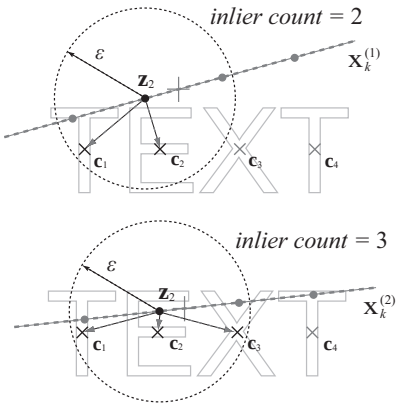


Figure 6. Inlier count of feature $\mathbf{z}_2$ for two different particles $\mathbf{x}_k^{(1)}$ and $\mathbf{x}_k^{(2)}$.

3.2.4 Bounding box computation - Bounding box computation is crucial towards next possible stages such as extraction, recognition or superresolution. Thus, it is important that it is as stable and tight as possible. Once a posterior state is established by the particle filter, each feature $\mathbf{z}_i \in \mathbf{Z}$ is assigned a *Most Likely Estimate* (MLE), that is the text component $\mathbf{c}_j \in \mathbf{y}_k$ that most likely matches it. In Figure 5, the MLE of each feature is marked with an arrow. Not all tracked features will have a MLE each frame, as sometimes they are not found due to blur, clutter or occlusion.

After perceptual text grouping, each observed text component belongs to a group, and thus the MLE of each tracker feature also belongs to a group. The *Most Likely Group* (MLG) of a feature is the group to which this feature's MLE belongs to. Given this, the tracker's bounding box is then obtained by joining the bounding boxes of its MLGs.

3.2.5 Tracker creation and removal - Trackers are dynamically created when new text is detected, and removed when their associated text entity can no longer be found. After the grouping stage, any text group detected is a potential text entity to be tracked. But some of these groups may belong to text entities already being tracked. The tracking stage identifies the tracked components in the image via the MLE and MLG mechanisms. After the tracking cycle, any unidentified remaining groups are passed to a new tracker. Newly created trackers must continuously track their text for a number of frames to be considered *stable*. Trackers that fail to comply with this are promptly removed. The tracker removal mechanism is very simple. After a consecutive number of frames without a match, the track is considered lost and removed. Should the same text entity then reappear, it will be assigned a new tracker.

4. Results

The system was tested on a variety of typical outdoor and indoor scenarios, e.g. a hand-held camera while passing shops or approaching notices, posters, billboards etc. We present here the results from four typical scenarios. The full video sequences along with other results, including a sequence from [13], are also available online[3].

The results shown are: Figure 7: 'BORDERS' - walking in a busy street with several shop signs overhead, Figure 8: 'UOB' - walking past a signboard including an occlusion in a highly textured scene background, Figure 9'ST. MICHAEL'S HOSPITAL' - a traffic sign with both bright and dark text, complex background and significant perspective change, and Figure 10: 'LORRY' - with text also undergoing viewpoint changes. All sequences were at 640×480 resolution recorded at 15 fps with a consumer grade photo/video camera (Nikon Coolpix P2).

Table 1 shows the performance of the algorithm for the different sample scenes on an Intel Pentium IV 3.2Ghz processor. The results show the performance of the text segmentation and grouping subsystem alone, and the whole tracking process. Text segmentation is very fast. When measured off-line, the system was able to compute the results faster than the actual frame rate of the sequences. With the tracking, the performance of the system is *close to 10 fps on average*, depending on the complexity of the scene, making it promisingly close to realtime. For a simple scene with little background and one 5-character word, the *system could track it effortlessly at 15fps*. While the particle filtering framework is relatively fast, the SIFT matching of features reduces the performance when the number of candidate regions is large, such as in very complex backgrounds, e.g. in Fig. 8. A greater number of false positives (due to the vegetation) produced during segmentation put more stress on the tracking stage, which however rejected these regions due to the instability and lack of longevity of their trackers. Notice also in Fig. 8, the tracker survives the occlusion by the lamppost.

4.1. Discussion

The focus of this paper has been on a framework to track text as robustly and continuously as possible, bearing in

[3] Please see `http://vision.cs.bris.ac.uk/texttrack/`

Figure 7. Example scene 1 - BORDERS - notice several BORDERS signs come along in the sequence.

Figure 8. Example scene 2 - UOB including occlusion, also with much other texture.

Figure 9. Example scene 3 - ST. MICHAEL'S HOSPITAL - two regions, dark over light and vice versa.

Figure 10. Example scene 4 - LORRY

Table 1. Performance of the algorithm in mean frames per second.

	Text segmentation	Full algorithm
Scene 1	31.9 fps	13.2 fps
Scene 2	21.3 fps	4.9 fps
Scene 3	30.7 fps	9.6 fps
Scene 4	32.0 fps	10.6 fps

mind that momentary loss of a text region is not disastrous in terms of recognition. Once stable tracking is obtained after a few frames, the motion information could be used for fronto-parallel recovery as well as generation of a super-resolution representation for better OCR, e.g. as in [12]. In our system, it is more likely that text is missed if it is at sharp perspective viewpoints, than for a non-text region to be tracked with significant stability. We had no such non-text cases, but even if there were, one can assume that OCR would reject it at the next stage.

Some shortcomings of our work are: (1) the robustness of our tracker improves further, in terms of dropping a track only to be picked up again instantly, when we use a more complex motion model, but this means we move further away from a realtime goal, (2) SIFT has limited robustness to viewpoint variations, so big changes of point of view will make the trackers lose the features, and it is by far the slowest part of the system, however we are at the time of writing experimenting with a new method, (3) Our results can not be claimed to be fully realtime, however we are near enough and believe we can achieve it in our future short-term work, (4) even though our few thresholds are fixed they naturally can affect the quality of the results; we aim to address these by applying learning techniques to automate the process where necessary.

5. Conclusion

In this paper we have presented a close to realtime technique to automatically detect and track text in arbitrary natural scenes. To detect the text regions, a depth-first search is applied to a tree representation of the image's connected components, where each leaf in the tree is examined for three criteria. Amongst these criteria is the use of the Eigen-Transform texture measure as an indicator of text. This stage of the algorithm detects both bright and dark text in a single traversal of the tree. Following perceptual grouping of the regions into text entities, particle filtering is applied to track them across sequences involving severe motions and shakes of the camera. We have established a significant framework and can start to improve its individual components in our future work to better our results.

Acknowledgements

Carlos Merino is funded by FIT-350300-2006-92 project from the Spanish Ministerio de Industria, Turismo y Comercio, through the European Regional Development Fund. This work was carried out partly at Bristol University and partly at Universidad de La Laguna and the Instituto Tecnológico de Canarias.

References

[1] *Proc. of the 1st Workshop on Camera Based Document Analysis and Recognition (CBDAR)*, August 2005.
[2] Special issue on camera-based text and document recognition. *International Journal on Document Analysis and Recognition*, 7(2–3), July 2005.
[3] P. Clark and M. Mirmehdi. Recognising text in real scenes. *IJDAR*, 4:243–257, 2002.
[4] P. Clark and M. Mirmehdi. Rectifying perspective views of text in 3d scenes using vanishing points. *Pattern Recognition*, 36(11):2673–2686, 2003.
[5] A. Doucet, J. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.
[6] R.-J. Jiang, F.-H. Qi, L. Xu, G.-R. Wu, and K.-H. Zhu. A learning-based method to detect and segment text from scene images. *Journal of Zhejiang University*, 8(4):568–574, 2007.
[7] M. León, S. Mallo, and A. Gasull. A tree structured-based caption text detection approach. In *Fifth IASTED VIIP*, 2005.
[8] H. Li, D. Doermann, and O. Kia. Automatic text detection and tracking in digital video. *IEEE-IP*, 9(1):147–156, 2000.
[9] J. Liang, D. Doermann, and H. Li. Camera-based analysis of text and documents: a survey. *IJDAR*, 7(2):84–104, 2005.
[10] R. Lienhart. Indexing & retrieval of digital video sequences based on text recognition. In *ICM*, pages 419–420, 1996.
[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.
[12] C. Mancas-Thillou and M. Mirmehdi. Super-resolution text using the teager filter. In *CBDAR05*, pages 10–16, 2005.
[13] G. K. Myers and B. Burns. A robust method for tracking scene text in video imagery. In *CBDAR05*, 2005.
[14] M. Pilu. Extraction of illusory linear clues in perspectively skewed documents. In *CVPR*, pages 363–368, 2001.
[15] M. Pupilli. *Particle Filtering for Real-time Camera Localisation.* PhD thesis, University of Bristol, October 2006.
[16] H. Shiratori, H. Goto, and H. Kobayashi. An efficient text capture method for moving robots using dct feature and text tracking. In *ICPR*, pages 1050–1053, 2006.
[17] A. Targhi, E. Hayman, J. Eklundh, and M. Shahshahani. The eigen-transform & applications. In *ACCV*, pages I:70–79, 2006.
[18] M. Wienecke, G. A. Fink, and G. Sagerer. Toward automatic video-based whiteboard reading. *IJDAR*, 7(2):188–200, 2005.
[19] L. L. Winger, J. A. Robinson, and M. E. Jernigan. Low-complexity character extraction in low-contrast scene images. *IJPRAI*, 14(2):113–135, 2000.
[20] A. Zandifar, R. Duraiswami, and L. S. Davis. A video-based framework for the analysis of presentations/posters. *IJDAR*, 7(2):178–187, 2005.
[21] Z. Zhu, F. Qi, M. Kimachi, and Y. Wu. Using adaboost to detect & segment characters in natural scenes. In *CBDAR05*, 2005.

Appendix B

Sensory substitution for visually disabled people: Computer solutions

This appendix includes the full text for the following article. The article is part of the PhD by publication:

Title	Sensory substitution for visually disabled people: Computer solutions.
Authors	Antonio Rodríguez-Hernández, Carlos Merino, Oscar Casanova, Cristián Modroño, Miguel Torres, Raquel Montserrat, Gorka Navarrete, Enrique Burunat and José Luis González-Mora
Type	Journal
Journal	WSEAS Transactions on Biology and Biomedicine
Year	2010
Volume	7
Number	1
Pages	1–10
ISSN	1109-9518
URL	http://www.scopus.com/inward/record.url?eid=2-s2.0-77950406568&partnerID=40

Sensory Substitution for Visually Disabled People: Computer Solutions

ANTONIO FRANCISCO RODRÍGUEZ-HERNÁNDEZ[1], CARLOS MERINO[1], OSCAR CASANOVA[1], CRISTIÁN MODRONO[1], MIGUEL ÁNGEL TORRES[1], RAQUEL MONTSERRAT[1], GORKA NAVARRETE[1], ENRIQUE BURUNAT[2] and JOSÉ LUIS GONZÁLEZ-MORA[1].

Department of Physiology[1] and
Department of Psycobiology[2]
University of La Laguna
S. Cristóbal de la Laguna, Tenerife, The Canary Islands
SPAIN
afrguez@ull.es http://www.nf.ull.es

Abstract: - Sensory substitution can be defined as a technical-scientific discipline which aims to provide sensory disabled people with information they cannot acquire from the disabled sense through their intact senses. We present here our team's work in this R+D line for providing blind and severe visually impaired people with real time spatial and text environmental information through sounds. The objective is to model the real environment as a virtual space where the object's surface appears as if covered by small sound sources, which emit very locatable sounds in a continuous and near simultaneous way. It is based on the hypothesis that the brain, when provided with this highly rich spatial information, will generate a kind of visual-like perception of the surrounding world. In this paper we describe our approach to this field and the main results obtained, which have practical consequences in the field of sensory rehabilitation as well as on the theory of perception.

Key-Words: Sensory substitution; Blindness; Sonification; HRTF; Computer vision; Brain plasticity.

1 INTRODUCTION

Once particular surrounding information has been identified as relevant for the sensory disabled persons, the sensory substitution approach firstly focuses on finding the optimal way of representing that information through the person's remaining senses and subsequently implementing it [1]. Applications of this concept range from traditional substitution methods like the long cane and Braille and Sign languages to the most recent developments based on high technology for acquiring and presenting the information of interest. In our approach, we use three technologies: computer vision, virtual reality and 3D sound.

Sensory substitution relies on the fact that environmental information is on many occasions available for a person throughout different sensory modalities. This is particularly certain for the spatial information. So, for example, the egocentric location of an object can be known by vision, audition, touch and even olfaction and the sense of temperature. This suggests that the brain may manage spatial information in an amodal way, that is, independently of the sensory modality that provides the information [2].

Several neurophysiologic studies support this notion. So, in the inferior Colliculus Nucleus of the Barn Owl, an early processing station at the auditory pathway of this predator species with an accredited ability for detecting and localizing relevant sounds, it has been found a topographic distribution of the auditory neurons sensitive to the location of the sound [3]. This distribution, although yet not found in higher cortical levels nor in humans, reminds the retinotopic organization of the visual pathway, where contiguous neuronal areas process contiguous areas of the perceptual field, showing that a very important feature for the visual spatial perception can be also developed for the auditory sense, in the form of a kind of "auditory retina"

The Posterior Parietal Cortex of the brain contains different areas. These areas are the 7ª area, the Lateral Intraparietal area (LIP), the Temporal Medial Superior area (MST), the 7b area and the Intraparietal Ventral area. In particular, the LIP area receives a considerable amount of projections from visual areas [4, 5], what justifies the notion of the LIP as the "parietal visual field". Nevertheless, the LIP area has

been later described as a receiver region of acoustic information which, in conjunction with the visual and the somatosensory one, contributes to generate the representational map of the three-dimensional space [6]. In this sense, it has been shown that the auditory response of the LIP neurons has the same preferential directionality than the visual response, what suggests that both sensory receptive fields, the auditory and the visual one, together with their respective sensory memory fields, overlap each other [7]. These multiple sensory inputs (visual, auditory, somatosensory and vestibular sense) to the LIP area are combined in a process of signal maximization for the coding of the spatial coordinates, what forms the basis of the surrounding spatial representations. In addition, these spatial representations at the posterior parietal cortex are related with high level neuronal cognitive activities, including attention.

Regarding sensory substitution, the cue point is to know whether the information required to carry out a particular perceptual task can be or not provided by one or more intact senses. In this sense, we have developed a series of prototypes for blind people's orientation, mobility and environmental perception, which provide the user with two types of real time information through sounds: information on the spatial volume occupied by the objects and surfaces located in front of him, and information on the written text present in the frontal scene (i.e., shop signs, advertisings, etc). The volume information is translated into a special sound code which is delivered through headphones in order to generate an auditory spatial representation coherent with the environment. The text information is presented as verbal spatial sound, as if a reading voice was coming from the area where the text is located. Then we follow a psychoacoustical approach in order to evaluate the users' perceptual response to the vision inspired acoustic stimulation. This work is complemented with the study of the disabled people's neurological substrate of the sensory substitution experience, through brain function registering techniques such as functional Magnetic Resonance Imaging (fMRI) and Event Related Potentials (ERP).

2 APPROACH

This section explains our approach to the problem of sensory substitution in blindness and severe visual impairment. We first pose our hypothesis regarding what particular information from the visual scene should, and then could, be acoustically provided to the listener so he will experience an auditory visual-like image of such scene. Next, the key methodological steps and the main results will be summarized. Finally we outline the current state of the on real time environmental text reader system.

2.1. The auditory code for the visual scene

A sighted person perceives images as series of light rays coming from every point of objects and surfaces inside his field of view. In a similar way, a person can also obtain a tactile sequential representation of the objects by touching them coordinate by coordinate. The perceived spatial image of an object comes from the acquisition of very significant spatial information, i.e., the subject centered spatial coordinates which are occupied by the objects. Following on from this idea, the following hypothesis can be posed: a blind person is exposed to sound rays radiating from an object's surface in such a way that his perceptual system can recover the whole set of relative spatial coordinates involved, similar to what happens in vision from perceiving light, will he be able to perceive some kind of crude visual-like image, similar perhaps to a 3D visual image containing mainly low spatial frequencies?

The literature reports several devices which offer auditory spatial information for the blind person's orientation and mobility (see [8] for a review on the earlier developments).These devices mainly provide auditory information that indicates the presence and location of the detected objects in order to avoid or to use them as landmarks when navigating. Dr. Kay's KASPA system is based on an ultrasonic sensor that calculates the obstacle distance from the time of flight of the emitted ultrasonic wave. This distance information is translated into a sound code which consists on a progressive change in the pitch of a pure tone as changes the detected distance. His system has later evolved to a version that includes wide-angle overlapping peripheral fields of view with a narrow central field superposed, what improves the blind subjects' auditory ability to resolve close objects [9]. Dr. Jack Loomis and his colleagues from UCSB were who first applied binaural technology for both representing the location of environmental landmarks and subsequently guiding the user's steps towards them. In this case, GPS information is translated into spatial verbal and non-verbal sound indications [10]. More recently, Dr. Tiponut and his team from the Polytechnic University of Timisoara have developed an integrated multisensory device which provides both information on the location of obstacles and a pilot signal to indicate the direction of the movement to a target. This information is coded as spatial sound obtained from binaural technology [11].

Some teams have also explored the human's ability to recognize an object's shape from both sound and tactile cues. Dr. Meijer's The Voice system directly translates the position and level of gray of the pixels composing the image from a video camera into a sound code where the vertical dimension is represented by the pitch of a pure tone, the horizontal dimension by both binaural cues and the presentation time, and the level of gray by the amplitude of the sound [12]. Dr. Capelle and colleagues from Catholic U. of Louvain have explored a similar shape codification by also attaching a pure tone to every pixel of the image from a video camera, although in this case the sound representing any activated pixel is emitted in a continuous way and only black and white levels of gray are considered [13]. Dr. Lakatos found that normally sighted subjects show considerable ability in recognizing alphanumeric characters whose patterns are outlined acoustically through the sequential activation of specific units in a speaker array[14]. Dr Hong and Dr. Beilharz, from the University of Sidney, find that the shape of two concurrent graphic lines can be gathered from an auditory representation based on mapping the x-axis to time and the y-axis to MIDI notes. The performance is improved when the concurrent audio streams are presented as independent separated virtual sound sources [15].

Regarding the tactile devices, it is mandatory to cite the work of Dr. Bach-y-Rita and colleagues who, in the 60s of the past century, introduced the concept of sensory substitution and explored the human ability to perceive object's shape and width from a bi-dimensional tactile projection of the scene onto the user's skin [16]. The image of single objects from a CCD camera is directly translated into a spatial pattern of vibro-tactile effectors which stimulate the corresponding coordinates on the user's skin. Later, their work derived toward the development of an electro-tactile interface placed on the tongue [17]. Drs. Segond, Weiss and Sampaio have recently explored its possibilities for perceiving shape and spatial cues for navigation [18].

The studies above suggest that spatial hearing, like vision and touch, has access to the mechanisms that give rise to the amodal spatial representations involved in the perception of the shape and width of the objects, that is, in the figure perception. In order to carry this argument one step further, we have explored what the perceptual effect of coding the environment with 3D sound is, i.e., coding the environment with sounds which are perceived coming from every occupied space coordinate.

In this vein, we have focused on whether an image of a unitary whole object, broadened in the space as in the visual experience, can be generated in the blind person from the appropriate auditory stimulus [19, 20, 21]. Figure 1 shows this approach: a sighted person sees the frame of a window as light rays coming from it. Then, a series of small loudspeakers are located occupying the same space location of the frame. We would expect the appropriate emission from that spatial configuration of loudspeakers to the blind person (third picture) to experience a spatial image of the frame which is spatially similar to the visual one.

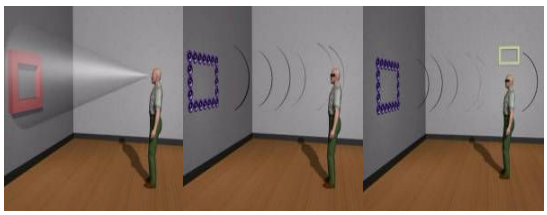


Fig.1. Visual to auditory sensory substitution

According to this scheme, we first explored blind people's ability to experience visual-like images from spatial patterns of real sound sources [19]. Figure 2 shows the experimental set-up consisting on a 6x6 array of small loudspeakers facing a point where the subject is placed to receive the emitted sound. Every loudspeaker is conveniently directed to the point where the subject's head will be placed, given that otherwise part of the higher spectral content of the sound would not reach the subject's ears, distorting the perceived elevation of the sound sources.

Fig.2. Loudspeakers array inside the acoustically prepared experimental room

These experiments were conducted with both blind and normally sighted people in an acoustically optimized environment. The set of loudspeakers configuring a particular spatial pattern is sequentially activated in order to test the subject's ability to recognize it as well as the spatial audio image being experienced. The presented figures are composed from, for example, the top and the bottom rows (two parallel horizontal lines), the left and the right columns (two parallel vertical lines), a C letter-form constructed with the left column and the top and bottom rows, the frame of a window, etc. The tests show that blind people can recognize the presented spatial patterns and clearly note its physical distribution, whilst referring that a kind of auditory spatial image extended in the space with the shape of the presented spatial pattern can be perceived.

Afterwards, a first augmented reality prototype was developed. This is a non portable laboratory prototype which first obtains the information of the spatial coordinates occupied by the objects in the scene, and then generates an auditory stimulus representing this information which is delivered to the user through headphones. This auditory stimulus is such that, in spite of being delivered through headphones, creates the illusion in the user that the previously detected object is covered by small emitting sound sources. This effect is obtained by combining computer vision techniques [22] for recognizing the environment with 3D sound techniques based on HRTF filtering (Head Related Transfer Function) [23, 24, 25], for creating the illusion of sound "externalization". This initial version sums up the philosophy of the subsequent prototypes, a brief description and the main results obtained are as follows:

2.1.1 The first device, Virtual Acoustic Space I: the validation of the idea.

VAS I consists of two subsystems: the first one for acquisition and analysis of the scene (visual subsystem), and the second one for conversion of the information into sounds and playing them back to the subjects (acoustic subsystem)[19].

Figure 2 shows a conceptual diagram of the technical solution we have chosen for the prototype development.

Two miniature cameras are fixed on both sides of a pair of conventional spectacles, which will be worn by the blind person using the system. Different computer vision algorithms are applied to the captured images, such as the detection of geometric features or stereovision, in order to obtain a depth map.

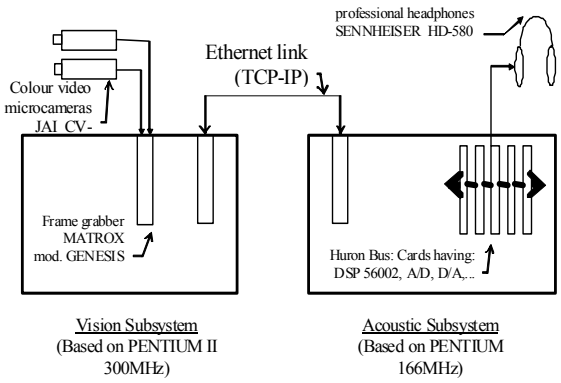


Fig.3. Prototype conceptual diagram

The acoustic subsystem then plays a random sequence of short sounds, one for each position provided in the depth map. Each sound has been previously "spatialized" so, in spite of being heard by headphones, it seems to come from a certain position in the environment.

The virtual sound generator uses the Head Related Transfer Function (HRTF) technique in order to obtain the spatial sounds [24]. For each position in space a set of two HRTFs are needed, one for each ear, so that the interaural time and intensity difference cues, together with the behavior of the outer ear are taken into account. In our case we are using a reverberating environment, so the measured impulse responses also include the information related to the echoes in the room. Individual HRTF's are measured as the responses of miniature microphones (placed at the entrance of the auditory channel) to a special measurement signal (MLS) [26]. Also the transfer function of the headphones is measured in the same way, in order to equalize its contribution.

Having measured those two functions, the HRTF and the Headphone Equalizing Data, properly selected or designed sounds can be filtered and presented to both ears, obtaining the same perception as if the sound source were placed in the position from where the HRTF was measured.

The sound selected for encoding the object's coordinates is a very short click without tonal quality. This type of sound is easily locatable in space and, given its short duration, it makes it possible to present a high number of coordinates within a short period of time. The perceptual effect of this stimulus could be described as hearing a large number of raindrops hitting the surface of a glass window. A field of view of 80° on the horizontal axis by 45° on the vertical axis is divided into in a number of x (horizontal), y (height) and z (distance) coordinates or stereopixels.

2.1.2. Results summary

The initial results were obtained in a controlled environment from a broad group of blind and visual impaired people [27], and they have been confirmed by multiple tests subsequently carried out with the more advanced versions of the device. These results support the hypothesis that, when using an auditory stimulus, as previously described, to represent large objects in a scene, it is possible to generate a perceptual experience in the user of a global and maintained presence of those different objects inside the field of view, which are perceived as occupying the space with their gross shape, dimensions and location. For example, two walls surrounding a path are perceived as sounding objects which are always present on both sides of the subject, with their vertical and depth dimensions. A central soundless space can be perceived, and the blind person can walk or "look" through it (Fig.2.).

Fig.4. A participant signals the limits of the auditory spatial image corresponding to a hole in the wall

In one of the tasks the blind person is asked to point to the figure and "draw it in the air" by moving the arm along the perceived extension of the sound image (Fig. 5).

Fig.5. The participant is asked to move her arm through the area where she is perceiving that the sound seems to be extended through

The arm is in a straight position and the person holds in the hand a magnetic sensor for registering the coordinates of the arm movement. In addition, a verbal description of the perceived extension of the sound image is collected for every figure.

Blind people can discriminate a line of sounds from a single point of sound, introducing the concept of persistent broadened sound. He or she can also recognize the horizontal, vertical or diagonal layout of this audible line by perceiving the extension of the spatial image of the line (Fig.6).

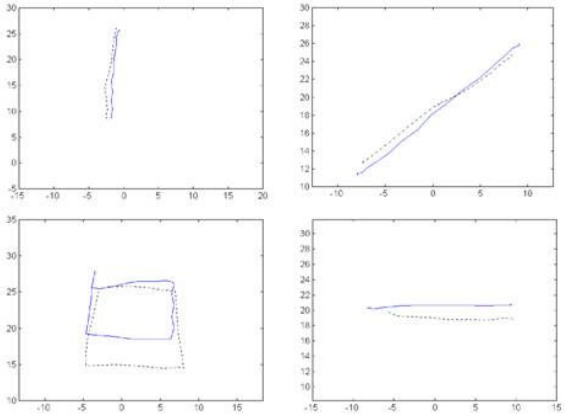


Fig.6. Register of the hand signaling of different figures from visual (continuous line) and auditory (dotted line) information (scale in inches)

These results are especially important in relation with the question of whether an auditory perception of objects similar to the visual one can be experienced or not. The scientific study of the spatial aspects of the auditory perception has mostly focused on the perception of isolated sound sources. This may be in relation with the fact that sounds are commonly perceived as coming from concrete isolated locations in the space. Nevertheless, sound sources are not single points but have a width. A spatial attribute of the spatial auditory image regarding to the "width of the sound" has been described, namely the sound broadness. Several acoustic parameters involved in this effect have been studied [28]. It is worth pointing out the references in these studies to the auditory spatial image that is experienced when hearing a multitude of close sound sources which are simultaneously radiating their respective sounds, for example, the rustling leaves and branches of a tree. Similarly, blind people who is asked about this question usually refers an auditory experience of sound broadness when perceiving the raindrops or the wind hitting diverse objects or surfaces in their surroundings. Somehow our approach operates on

these considerations and the results support the notion of auditory figure perception.

Finally, it is worth pointing out the experience of a blind person who was able to detect and identify the different objects and surfaces presented to her when using the prototype inside an experimental room and without any previous knowledge of this environment (walls, a column, a window, the door and a small table). She was able to move between them, and to make a correct verbal and graphical description of the room and the relative position of every object and surface.

As regards the neurological substrate of this perceptual activity, preliminary results using the functional Magnetic Resonance Imaging technique (fMRI) have shown that spatial sound processing in blind people occurs more in occipital cortical brain regions than in sighted people [29]. This suggests that blind people recruits the brain's visual areas for spatial sound processing, which has important consequences when considering the blind person's potential ability to use sensory substitution devices. Many other studies support also the notion of brain intermodal sensory plasticity (see for example [30]).

2.1.3 Recent work

The latest version of the prototype has been developed within a recent EUFP6 project whose name is CASBliP (Cognitive Aid System for Blind People-www.casblip.com). The CASBliP device acquires environmental information from two different independent subsystems: 1) a time-of-flight infrared sensor placed on the frame of a pair of glasses which acquires distance information from the objects in the frontal scene inside a range of 0.5m to 5m for a horizontal plane at eye level; and 2) a pair of cameras placed at the top of a helmet. Segmentation and shape identification algorithms enable us to detect a moving object in a range of 5m to 15m. This information is presented in an auditory way basically according to the representation strategy outlined above. A series of tests have been conducted which show that a blind person can perform orientation and mobility tasks with a progressive improvement with learning (Fig.7).

Here it is shown the performance (measured in seconds by meter) of ten blind persons who were asked to navigate, relying only on the sensory substitution portable system, through a 14 m long route, i.e. a path with 4 pairs of soft obstacles of 180 cm height put up asymmetrically (Fig.8).

The participants had to locate every pair of columns (1 m of separation between them) and detect the gap through which they might move without touching or knocking the objects over, and then go on to the next pair of columns.

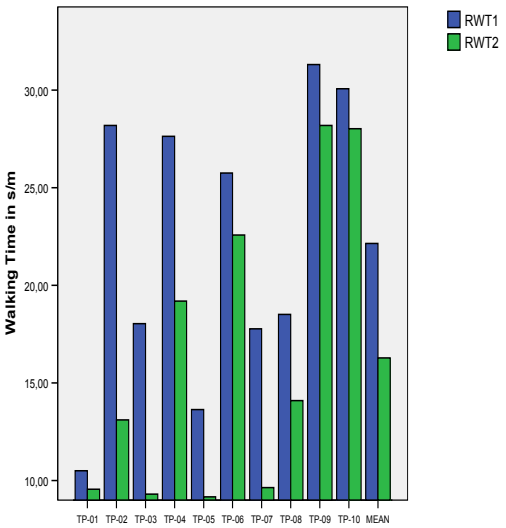


Fig.7. Relative Walking Time (s/m) of ten blind persons who were asked to navigate through the obstacles shown in Fig.5. In blue the results of the first run test (RWT1) and in green the results of a second run test (RWT2) after a short period of intensive training

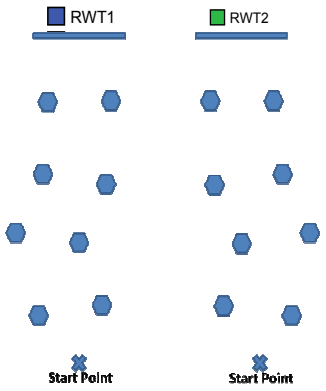


Fig.8. 14 m long route with 4 pairs of soft obstacles of 180 cm height put up asymmetrically. A flat surface indicates to the participant where the end of the course is. The differences between both arrangements try to avoid a possible influence of memory on the performance of the orientation and mobility task

The subjects had previously spent three sessions in order to become familiarized with the substitution stimulus, what required a mean time per subject of 75

min. After the first run test persons spent some time on the site with the mobility instructor to do training before they commenced a second test run, which one was carried out on a different arrangement of the obstacles. Relative Walking Time (RWT) scores from the post-training session show that results had significantly improved (t=4.36; p=0.002) [31].

A virtual reality simulator for blind and visually impaired people has been developed. It is called Virtual Reality Simulator for Sonification Studies, or VRS3 [32], and provides the user with a spatial auditory representation of the virtual environment previously designed. A 3D tracking system locates user's head orientation and position, so the user can "walk through" the virtual environment while he or she perceives the environment through auditory information (Fig.9).

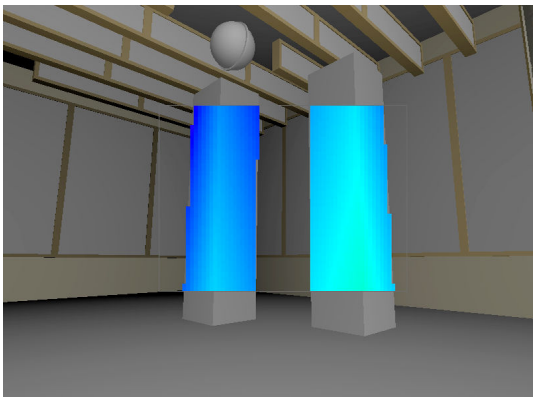


Fig.9. The depth map of two virtual columns inside the simulator room is overlapped to the scene and represents a 64x48 pixels size map with a pseudo-color depth scale

The simulator has these main purposes: validation of sonification techniques, 3d sensor emulation for environment recognition and hardware integration; also for training and auditory perception experiments. This simulator can recreate any simple or complex scene and present it to the user as a 3D sound world. Then, it allows the researcher to surpass the need of a "sensor system" for studying the perception of the auditory representation of the scenes. Concretely, we use it for defining the representation strategy, that is, the way the scene information is coupled to a sound code. Then the studies are oriented to get a better understanding on the perceptual effects of several significant acoustic parameters, as the interclick time interval, the sound reverberation level or the tonal colour of the click sound.

We also study the effect of training on the quality of the auditory spatial image of the scene experienced by the user, which implies some research on learning protocols of sensory perception as well as researching on individual or group differences (by sex, age, and so). In this way we have obtained preliminary results showing some advantages of more complexes (spectrally rich) sounds for distance localization using a set of real sound sources located from 50 cm till 6 meters in front of the subject (submitted for publication).

A robotic system has been developed that allows intensive measurement of both human and mannequin HRTFs in every spatial axe inside an 8 x 4 x 4 m width acoustically isolated room, with a spatial resolution up to 1º. This system allows getting massive sets of spatial filters from both subjects and mannequins. We are currently studying the effects of training on the precision of auditory localization when using both individual and generic virtual spatial sounds.

2.2 Development of a text reading system

The objective of this research line is to develop a scene text reading system for blind people. It is widely accepted that Optical Character Recognition (OCR) for scanned documents is no longer a problem. There are several commercial and open source OCR engines available, with recognition rates of over 95% for clean, scanned documents. Text recognition of scene text extracted from a video camera is a much harder problem and remains largely unsolved. There has been great interest in recent years in this field among research groups all around the world. Some applications are automatic indexing and cataloging of video libraries, road sign driver assistance, mobile phone document scanning, or visually impaired assistance systems, etc. Advances in digital cameras, computing power and modern computer vision techniques are making real-time text extraction and document processing from video images and its application on blind people assistance possible.

A system is being developed that detects, segments and tracks scene text such as shop signs, traffic signs, advertisements and billboards in nearly real-time (Fig.10).

For demonstration purposes, a simple communication module with an OCR engine and a voice synthesizer were integrated into the system [33].

Fig.10. Example of a text both detected and recognized by the portable version of the reading system (green letters inside the white box)

3 CONCLUSION

We present in this paper our R+D line in the field of sensory substitution for providing blind and visually impaired people with relevant visual environmental information through sounds. We mainly focus on exploring the hypothesis that perception of objects' spatial attributes like shape, width and location, can be experienced by the blind users in a gross visual-like way through hearing, whenever the appropriate acoustic representation of those spatial features is provided. Our results support this idea and encourage to going on in order to define optimal acoustic representations of the real scene information. In addition, a series of progressively more sophisticated prototypes has been developed with the aim of obtaining a portable device susceptible to be added to the existing arsenal of rehabilitation aids for orientation, mobility and perception of the environment.

Our group is currently continuing on with the improvement of the above-mentioned prototypes. We aim to develop an integrated portable prototype capable of acquiring, from the user's frontal scene, a robust 3D depth map segmented into objects, distant text information and both the identification and position of selected items, to immediately deliver this information as an adequate auditory representation based on spatial sound. The development of computer vision algorithms for video image segmentation, detection and labeling of the environment will enrich the depth map information provided by the 3D sensor. Concomitantly we study how different acoustic parameters affect the user's spatial auditory image of the scene, in order to optimize his or her auditory representation of it. The question of using individual versus generic or semi-personal HRTFS in order to achieve an appropriate spatial sound perception is still unsolved. In this sense, we are currently addressing the role of learning on the calibration of the auditory system to a non individual collection of spatial sounds.

Furthermore, the fact that blind people occasionally perceive spots of lights located at the spatial location of suddenly presented noises (which were reported as phosphenes in the decade of the 70's of the last century [34,35]) points to the fact that the brain can mixture sound and vision in a unknown way. Then we are currently researching to elucidate the neurological substratum of the phosphene phenomenon (visual perception elicited by sound stimulation), and preliminary results have already been reported [36].

We feel that the sensory substitution approach, when supported by the advances in high technology and a progressively better knowledge of the human brain's perceptual capabilities, opens up a wide field of applications in sensory rehabilitation.

References:

[1] Bach-y-Rita, P. "Brain mechanisms in sensory substitution". New York: Academic Press.

[2] Loomis, J. "Sensory replacement and sensory substitution: overview and prospects for the future". Converging technologies for improving human performance. Edited by Mihail C. Roco and William Sims Bainbridge. Kubler Academic Publisher, 2003.

[3] Takahashi, T. T., Keeler, C.H. "Representation of multiple sound sources in the owl´s auditory space map". Journal of Neuroscience. Vol 14, No 8, 4780-4793. August, 1994.

[4] Lynch J.C., Graybiel A.M. Lobeck L.J. "The differential projection of two cytoarchitecture subregions of the inferior parietal lobule of macaque upon the deep layers of the superior colliculus". J. Comp. Neurol. 235:241-254. 1985.

[5] Blatt G.J. Andersen R.A. Stoner G.R. "Visual receptive-field organization and cortico-cortical connections of the lateral intraparietal area (lip) in macaque". J. Comp Neurol. 299: 241-245. 1990.

[6] Mazzani P., Bracewell, R.M., Barash, S., Andersen, R.A. "Spatially tuned auditory responses in area LIP in macaques performing

delayed memory saccades to acoustic targets". J. Neurophysiol. 75: 1233-41. 1996.

[7] Andersen, R. A. "Multimodal representation of space in the posterior parietal cortex and its use in planning movements". Annual Review Neuroscience. Vol 20, 303-330. 1997.

[8] Kay, L. "Electronics aids for blind persons: an interdisciplinary subject". IEE review. IEE Proceedings, Vol. 131, Pt. A, No. 7, 559-576. September 1984.

[9] Kay L. Auditory perception of objects by blind persons, using a bioacoustic high resolution air sonar. J Acoust Soc Am. 2000 Jun;107(6):3266-75.

[10] Loomis, JM. Klatzky RL, Golledge RG. "Navigation without Vision: Basic and Applied Research". Optometry and Vision Science, Vol. 78, No. 5, May 2001.

[11] Virgil Tiponut, Zoltan Haraszy, Daniel Ianchis, Ioan Lie. "Acoustic Virtual Reality Performing Man-machine Interfacing of the Blind". 12th WSEAS International Conference on SYSTEMS, pp 345-349, Heraklion, Greece, July 22-24, 2008.

[12] Meijer, P.B.L. "An experimental system for auditory images representations". IEEE Transactions on Biomedical Engineering, 39; 112-121. 1992.

[13] Capelle, C.H. et al. "A real time experimental prototype for enhancement of vision rehabilitation using auditory substitution". IEEE Transactions on Biomedical Engineering. Vol. 45, nº 10, pp 1279-1293. Oct. 1998.

[14] Lakatos, S. Recognition of complex auditory spatial patterns. Perception. Vol. 22. 363-374. 1993.

[15] Hong Jun Song, Kirsty Beilharz. "Spatialization and Timbre for Effective Auditory Graphing". Proceedings of the 8th WSEAS International Conference on Acoustics & Music: Theory & Applications, pp 18-26, Vancouver, Canada, June 19-21, 2007.

[16] Bach-Y-Rita, P. et al. "Visual substitution by tactile image projection". Nature, 1221, 963-964. 1969.

[17] Bach-y-Rita, P., Kaczmarek, K.A., Tyler, ME, García-Lara, J. "Form perception with a 49-point electrotactile stimulus array on the tongue: a technical note". J Rehabil Res Dev. Oct; 35(4):427-30. 1998.

[18] Segond, H., Weiss, D., Sampaio, E. "Human spatial navigation via a visuo-tactile sensory substitution system". Perception. 2005; 34(10):1231-49.

[19] Rodríguez-Ramos, L.F., Chulani H.M., Díaz-Saco L., Sosa N., Rodríguez-Hernández A., González Mora J.L. "Image And Sound Processing For The Creation Of A Virtual Acoustic Space For The Blind People". Signal Processing and Communications, 472-475, 1997.

[20] González Mora, J.L., Rodríguez-Hernández, A., Rodríguez-Ramos, L.F., Díaz-Saco, L., Sosa, N. "Development of a new space perception system for blind people, based on the creation of a virtual acoustic space". Proceedings of the International Work Conference on Artificial and Natural networks. Vol. 2, pp 321-330. Springer. 1999.

[21] Rodríguez Hernández, A; Sosa, N, Rodríguez Ramos, L.F. , Chulani, H. Díaz Saco, L, y González-Mora, J.L.. "Percepción del entorno en personas ciegas a través de un estímulo sonoro espacial virtual generado por computador". Actas del Congreso Iberoamericano 3º de CAA, 1º de Tecnologías de Apoyo para la Discapacidad, Oct. 2000. pp 81-84. Ramón Cerés y Comité Organizador Editores.

[22] Faugeras. "Three dimensional computer vision. a geometric viewpoint". The MIT Press. 1993.

[23] Begault, D. R. "3-D sound for virtual reality and multimedia". 1994. AP Professional.

[24] Wightman, F., Kistler, D.J. "Headphone simulation of free field listening I: stimulus synthesis". Journal Acoustical Society of America. V. 85 858-867. 1989.

[25] Kenneth John Faller II, Armando Barreto, Navarun Gupta And Naphtali Rishe. "Enhanced modeling of head-related impulse responses towards the development of customizable sound spatialization". 4th WSEAS Int. Conf. on Computational Intelligence, Man-Machine Systems and Cybernetics, pp 82-87, Miami, Florida, USA, November 17-19, 2005.

[26] Rife, D., Vanderkooy, J. Tranfer-function measurement with maximum-length sequences. J. Audio Eng. Soc., Vol. 37, No. 6, 1989.

[27] González-Mora, J.L., Rodríguez-Hernández, A.F., Burunat E., Chulani, H.M, Albaladejo, J.C., Rodríguez-Ramos, L.F. "Seeing the world by hearing: Virtual Acoustic Space (VAS) a new space perception system for blind people." Ballesteros, S., & Heller, M. A. (Eds.). pp 371-383. UNED. 2004.

[28] Blauert, J., Lindemann, W. "Auditory spaciousness: some further psychoacoustic analyses".J. Acoust. Soc. Am. Aug; 80(2):533-42. 1986.

[29] Rodríguez Hernández, A.F., González Mora, J.L., Pujol, J., López García, J.A.,Muñiz Montes, J.R., Rodríguez Ramos, L.F., Chulani, H.M. Burunat, E. "Functional magnetic resonance imaging of spatial auditory perception in blind

and sighted human subjects". J Physiol. 548P O76. 2003.

[30] Sadato, N. Pascual-Leone, A. Grafman, J., Ibáñez, V., Daiber, M.P., Dold, G., Hallett, M. "Activation of primary visual cortex by braille reading in blind people". Nature. 380,526-527. 1996.

[31] Hans Kaltwasser. Internal Document CASBliP Project. Deutscher Blinden-und Sehbehinder-tenverband e.V. (DBSV), 2008.

[32] Miguel Ángel Torres Gil, Óscar Casanova González, and José Luis González-Mora. "Virtual reality simulator for sonification studies". Recent Advances in Computational Intelligence, Man-Machine Systems and Cybernetics, Proceedings of the 8th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics (CIMMACS '09), pp 112-116. El Puerto de la Cruz, Tenerife, December 14-16, 2009.

[33] Merino, C., Mirmehdi, M. "A Framework Towards Realtime Detection and Tracking of Text". In: Second International Workshop on Camera-Based Document Analysis and Recognition (CBDAR 2007), pp10–17. Sept 2007.

[34] Lessell, S. and Cohen, M.M. "Phosphenes induced by sounds". Neurology, 29(11): 1524-1526. 1979.

[35] Page, N.G.R., Bolger, J.P., Sanders, M.D."Auditory evoked phosphenes in optic nerve disease". Journal of Neurology, Neurosurgery and Psychiatry. V. 45, 7-12. 1982.

[36] Burunat E., Rodríguez-Hernández A. F., López-García J. A., Muñiz-Montes J. R., Pujol J., Rodríguez-Ramos L. F., Chulani H.M., González-Mora J. L. "Blind woman experiencing auditory evoked phosphenes: A functional magnetic resonance imaging (fMRI) study". FENS Abstr., A224.3, vol.2, 2004.

Acknowledgements

This R+D line has been carried out with funds from the Spanish MCyT, MCeI and MITyC, the Govern of Canarias, the Spanish National Blind Organization (ONCE), EU-FEDER Program, EUFP6-IST, Agencia Canaria de Investigación, Innovación y Sociedad de la Información (ULLAPD-08/01), Ministerio de Industria, Turismo y Comercio (Proyecto AVANZA), TSI-020100-2008-337; Ministerio de Ciencia e Innovación (Proyecto CICYT), TIN2008-06867-C02-01/TIN. We express our gratitude to Hans Kaltwasser, from DBSV, for sharing partial results of our joint work in the CASBliP Project and to Manuel Mauricio, from ULL, for the infographics of figure 1.

Appendix C

A Head-Mounted Device for Recognizing Text in Natural Scenes

This appendix includes the full text for the following article:

Title	A Head-Mounted Device for Recognizing Text in Natural Scenes.
Authors	Carlos Merino-Gracia Karel Lenc and Majid Mirmehdi
Type	Conference proceedings
Conference	Camera-Based Document Analysis and Recognition
Series	Lecture Notes in Computer Science
Editors	Masakazu Iwamura and Faisal Shafait
Publisher	Springer Berlin / Heidelberg
Year	2012
Volume	7139
Pages	29–41
ISSN	0302-9743
DOI	10.1007/978-3-642-29364-1_3
URL	http://dx.doi.org/10.1007/978-3-642-29364-1_3

A Head-Mounted Device for Recognizing Text in Natural Scenes

Carlos Merino-Gracia[1], Karel Lenc[2], and Majid Mirmehdi[3]

[1] Neurochemistry and Neuroimaging Laboratory, University of La Laguna, Spain
cmerino@ull.es
[2] Center for Machine Perception, Czech Technical University, Czech Republic
lenckar1@fel.cvut.cz
[3] Visual Information Laboratory, University of Bristol, UK
majid@cs.bris.ac.uk

Abstract. We present a mobile head-mounted device for detecting and tracking text that is encased in an ordinary flat-cap hat. The main parts of the device are an integrated camera and audio webcam together with a simple remote control system, all connected via a USB hub to a laptop. A near to real-time text detection algorithm (around 14 fps for 640×480 images) which uses Maximal Stable Extremal Regions (MSERs) for image segmentation is proposed. Comparative text detection results against the ICDAR 2003 text locating competition database along with performance figures are presented.

Keywords: wearable device, text detection, text understanding, MSER.

1 Introduction

The area of wearable computing has seen relatively little growth over the last few years after the initial wave of enthusiasm in the area, mainly due to the miniaturisation of personal computing devices, such as mobile phones that need not be worn, but carried, that perform most of our everyday needs. Also, the focus of recent advances in wearable computing have been in specific and specialist areas, e.g. in health monitoring systems. Regardless of this, wearable devices for everyday and general purpose use are still extremely important to help those most in need of it, e.g. disabled users such as the blind, or those incapacitated by language barriers, e.g. tourists!

In this work, we present a simple hat, with embedded camera, speaker, and USB port (see Fig. 1) for an application that involves the real-time detection and tracking of text. The camera provides real-time video, via a discreetly hidden USB cable, to a small laptop (to be carried) where the number crunching occurs. The results of text detection and recognition is returned to the hat via an audio signal on the USB port to a speaker embedded in the hat (which can be used with earphones if necessary). All electronic components are off-the-shelf and are held in a part which is readily removable from the hat. This allows us to easily extend the device in the future just by adapting the removable part, for example

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 29–41, 2012.

(a) (b)

Fig. 1. Developed device together with remote control (a-*i*) and its shape when used (a-*ii*). Removable part (a-*iii*) is placed inside a hat (a-*iv*) in a metal framework which is visualized in image (b). The device comprises a USB camera with auto focus (1), a RC receiver (2) and a USB sound card (3) which are connected to a USB hub (4).

with an embedded computer which will be able to handle all the computation. Since the device does not require units integrated with shades or spectacles, it does not interfere with users who have some residual vision.

Helping visually impaired people to understand the scene in their surrounding environment is a major goal in computer vision, with text detection and its communication to the user a significant aspect of it. One of the earliest approaches can be considered to be the assistive technology approach by Kurzweil's reading machine [8] in 1975 which enabled book reading for blind people using a flat CCD scanner and computer unit with optical character recognition (OCR) and text to speech synthesis (TTS) systems. Several desktop solutions with a similar design are still widely available. This layout was improved using a camera, for example in the iCARE portable reader [7] which made document manipulation less cumbersome. In Aoki et al. [1], a small camera mounted on a baseball cap was used for user navigation in an environment. Chmiel et al. [2] proposed a device comprising glasses with integrated camera and DSP-based processing unit which performed the recognition and speech synthesis tasks. However, this device was directed mainly towards document reading for the blind. The SYPOLE project [21] designed a tool primarily intended for reading text in the user's natural environment by taking snapshots of documents, e.g. banknotes, via a camera mounted on a hand-held PDA device.

In the context of other application areas, detecting and recognizing text is important for translation purposes, e.g. for tourists or robots. This is a subject of interest for the Translation robot [22] which consists of a camera mounted on reading glasses together with a head-mounted display used as the output device for translated text.

Text detection has received increasing attention in recent years, with many works surveyed in [9] and [24]. An example of a recent approach is Pan et al. [20]

who combined classic region-based and connected components-based (CC) methods into a complex text detection system and achieved the best results on the ICDAR dataset yet (used for performance measurement by many text detection algorithms). Their system binarized the image in the first stage based on a text confidence map, calculated from classified gradient features of different sized regions. Segmented CCs were then classified using learned condition random field parameters of several unary and binary component features. Another recent example is Epshtein et al. [5] who used the stroke-width transform to obtain candidate text regions formed of CC pixels of similar stroke widths.

Contrary to the degree of attention enjoyed by text detection, text tracking has been hardly investigated considering that it is very important for reasonable user interaction in any text detection system involving ego or object motion. In our previous work [15], we developed a real-time probabilistic tracker based on particle filtering which is used in the proposed text detection system here. We are only aware of one other work, Myers and Burns [16], who tracked text by feature correspondence across frames by correlating small patches. While we have developed our text tracking application beyond what we previously reported in [15], the focus of the work presented here is on text detection and on the hat-based communication device. Our most recent results on text tracking will be presented in a future work.

In this paper we also examine the use of Maximally Stable Extremal Regions (MSER) [13] for text detection. Originally developed as a method to detect robust image features, the method responds well to text regions. MSER has been used for license plate detection [4] and more recently, Neumann and Matas [17] used MSERs in a supervised learning system for text detection and character recognition using SVM classifiers. Although this method yielded promising results it is computationally expensive. Our approach is based on MSER as a candidate text region detector but we rely on the hierarchical relationship between detected MSERs to quickly filter through them (Section 3). Then a cascade of text classifying filters is applied to candidate text regions. Using much simpler text classification techniques allows us to provide a close to real-time implementation. We present single image text detection performance results evaluated against the standard ICDAR 2003 text locating competition database. Performance figures are provided to illustrate the efficiency of the algorithm (Section 4)[1].

2 Hardware Design

Placing a camera in a hat is a logical choice as it is both an unobtrusive location and an ideal position in reference to where the eyes and head point to. Mayol et al. [14] examined possible positions of wearable cameras and concluded that head mounted cameras provide the best possible link with the user's attention.

[1] Additionally, example videos recorded using the hat can be downloaded from: `http://www.cs.bris.ac.uk/home/majid/CBDAR/`

The hardware proposed here allows its integration into many varieties of hats, here we have used an ordinary fashion accessory – a *flat-cap*.

The hardware was developed with emphasis on robustness, serviceability and visual appearance. Figure 1a shows the appearance of the completed device. It is composed of a fixed part and a removable one. The fixed part is an aluminium plate, bent into a shape that very loosely follows the curves of the hat, while protecting the space used by the removable part. It has an opening in the front side, protected by a glass cover, which fits onto the camera lens. This provides dust and, to some degree, weather insulation. The removable part holds all the electronic devices. It is built out of commodity hardware, with a total cost of all the components under 100€: a high definition web camera with adjustable focus (Logitech Quickcam Pro 9000), an USB sound card used for voice feedback to the user through a pair of connected headphones, a RF transceiver and an USB hub. A view of the inner part of the hat, with the removable part and the electronic parts is shown in Fig. 1b.

The device is controlled by a hand-held remote control which acts like an ordinary USB keyboard. To minimize the number of cables, all the devices are connected to a generic USB hub which allows connecting the hat to any USB-enabled *computing device*, from tablets to fully equipped laptop computers, with a single cable.

3 Proposed System

A simplified schematic of the proposed system is shown in Fig. 2. Initially, we detect candidate text regions in the image input stream using our MSER-based approach. These regions are then tracked in consecutive frames and are eventually analysed using the open source Tesseract OCR[2] engine integrated into our software. Recognized text regions above a significant confidence measure determined by the OCR engine are then sent to a text-to-speech synthesis module (Flite TTS[3], also integrated into our software).

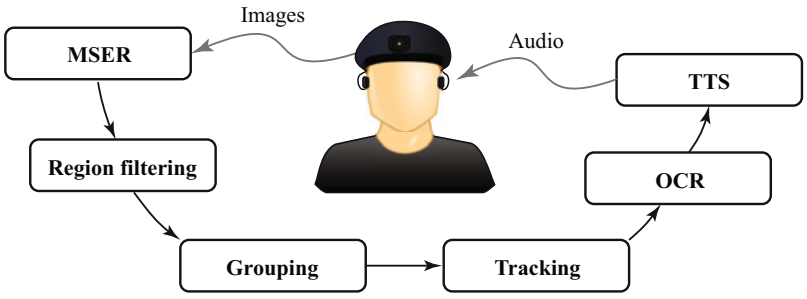


Fig. 2. General structure of the text detector application

[2] Tesseract OCR: `http://code.google.com/p/tesseract-ocr/`
[3] Flite TTS: `http://www.speech.cs.cmu.edu/flite/`

In our previous work on text detection and tracking [15] we used adaptive thresholding to initially binarize the original image. Then a tree was constructed representing the topological relationship between CCs in the binary image. A key step of the algorithm was a hierarchical filtering of the tree nodes, which allowed the rejection of many candidate regions without classification. After that, the remaining tree nodes where filtered using a cascade of text classifiers.

The approach proposed here uses Maximally Stable Extremal Regions [13] for image segmentation along with hierarchical filtering similar to our previous work.

3.1 Image Segmentation

MSERs are regions of interest in an image which present an extremal property of the intensity function around its contour. When applying a varying threshold level to a grey scale image, CC regions in the thresholded image evolve: new regions appear at certain levels, regions grow and eventually join others. Those regions which keep an almost constant pixel count (area) for a range of threshold levels are called MSERs. This technique, originally proposed as a distinguished region detector, also presents very desirable properties when applied to text detection, such as stability and multiscale detection.

MSERs can also be obtained by filtering the *component tree* of the source image, as shown by Donoser et al. [3]. The component tree is a representation of all the CCs which result from applying a varying threshold level to a grey scale image. The CCs are laid out in a hierarchy representing the topological relationship between them. A stability factor – i.e. the rate of change in the area of the components – is computed for each node in the component tree. MSERs are identified as local minima of the stability factor along paths in the tree towards the root.

We use the efficient, linear time MSER algorithm by Nister et al. [18], which crucially also constructs the component tree. We make two passes on the original image. First, MSER+ regions are obtained by applying the MSER algorithm on the image. This produces light regions inside dark ones. Then MSER- regions are obtained by applying the MSER algorithm to the inverse (negative) of the original image which produces dark regions inside light ones. The sets of regions returned by each pass are disjoint and both passes are needed to detect light text on dark backgrounds and dark text on light backgrounds. The algorithm can be easily modified to return a *hierarchical MSER* tree; an example output can be seen in Fig. 4b where blue regions were obtained by the MSER+ pass and the red regions by the MSER- pass. Darker regions represent upper tree nodes (closer to the root), while brighter regions show lower nodes (closer to the leaves). With hierarchical MSER, we have the desirable properties of MSERs as a distinguished region finder applied to text detection. Additionally, we keep the topological relationship of the CCs, which provides context information for later text filtering stages.

The resulting hierarchical MSER tree is then pruned in two stages: (1) reduction of linear segments and (2) hierarchical filtering. The first stage identifies

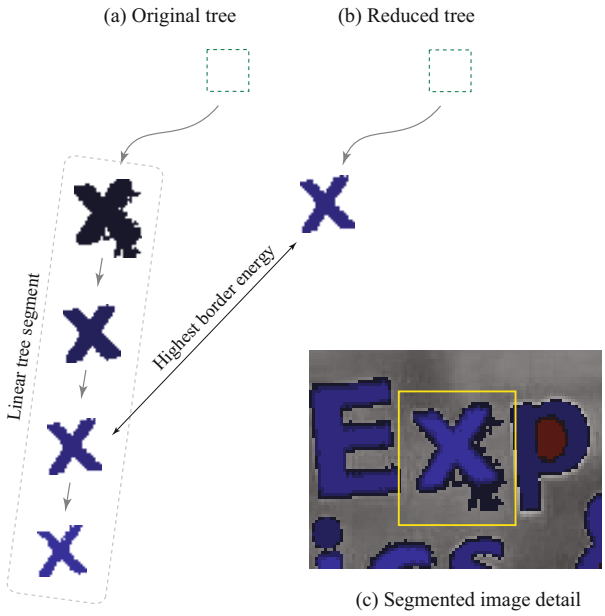
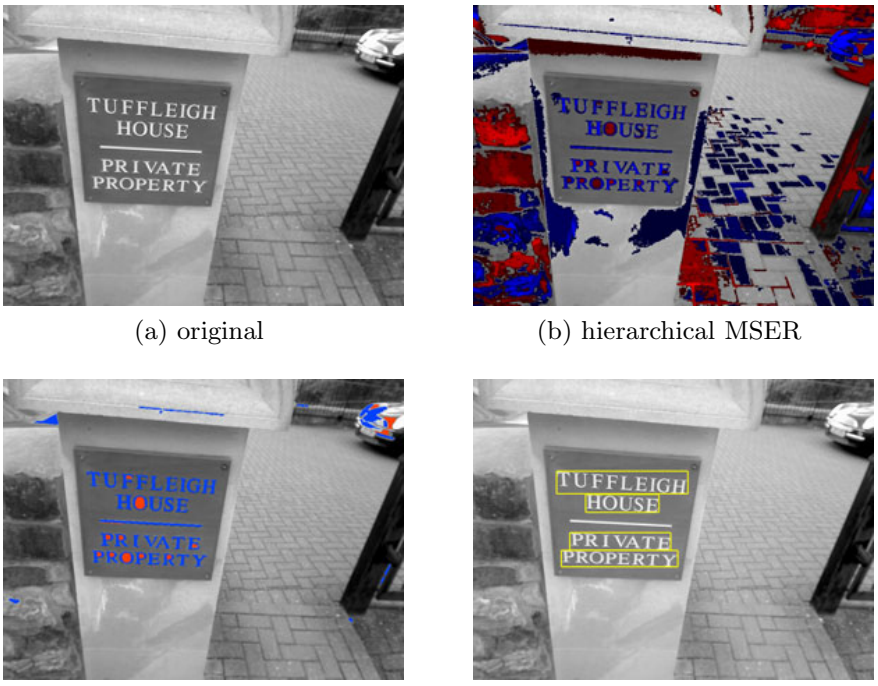


Fig. 3. Linear tree segments removal



(a) original (b) hierarchical MSER

(c) filtered MSER (d) grouping results

Fig. 4. Output from different stages of the text detection algorithm

all the linear segments within the tree where a linear segment is a maximum path between two tree nodes without any branches in between. Likewise, it is a path starting with a node with only one child, and ending with a branch node (a node with more than one child), or a leaf. Each linear segment is then collapsed into the node along the path, as shown in Fig. 3, which maximizes the *border energy* function (see below). In the second stage, the tree is walked depth-first, and a sequence of text classifying filters is applied to leaf nodes. Any non-leaf node without any descendant node classified as text is also tested with the text classifying filters. This stage is similar to the hierarchical tree filtering we originally proposed in [15]. Figure 4 shows the output of several stages of our text detection algorithm.

3.2 Region Filtering

During the tree walk, candidate regions are passed through a cascade of filters, i.e. *size*, *aspect ratio*, *complexity*, *border energy* and *texture*. This arrangement means that most of the regions will be discarded by the simpler filters, thus reducing the number of regions examined by the more complex tests. Thus, for a region i:

Size – the simplest condition filters out regions whose boundary length falls outside an allowed interval $(l_{min}; l_{max})$. The interval limits are fixed as a function of the image size:

$$l_{\min} < |\mathbf{B}_i| < l_{\max} \tag{1}$$

where $\mathbf{B}_i$ is the set of points around the region's boundary.

Aspect Ratio – given width W_i and height H_i of candidate region i, this condition rejects regions that are too wide or too narrow:

$$a_{\min} < \frac{W_i}{H_i} < a_{\max} \tag{2}$$

Complexity – this is a simple measurement of region complexity. It measures the ratio between the region boundary length and its area A_i. This criterion filters out regions with a rough border, which are usually produced by noise:

$$\frac{|\mathbf{B}_i|}{A_i} < c \tag{3}$$

Border Energy – this is a measure of contrast against the background. It filters out regions with low average edge response (from a Sobel operator (S_x, S_y)) around its boundary set of points $\mathbf{B}_i$, i.e. the region is valid only if its border energy exceeds a threshold:

$$\frac{1}{|\mathbf{B}_i|} \sum_{(x,y)\in\mathbf{B}_i} \sqrt{(S_x(x,y)^2 + S_y(x,y)^2)} > e \tag{4}$$

Texture Measure – the last filter in the sequence is a measurement of texture response, as text regions usually contain high frequencies. We found that the LU

transform [23] yields good response results when applied to text regions. It is a simple transformation based on LU decomposition of square image sub-matrices A around each interest point.

$$A = P\ L\ U \tag{5}$$

where L and U are lower and upper diagonal matrices and the diagonal elements of L are equal to one. Matrix P is a permutation matrix. In the LU decomposition, the number of zero diagonal elements of U is in direct proportion to the dimensionality of the null-space of A.

The actual texture response $\Omega_p(l, w)$ is calculated as the mean value of the diagonal values of the U matrix.

$$\Omega_p(l, w) = \frac{1}{w - l + 1} \sum_{k=l}^{w} |u_{kk}|, \qquad 1 < l < w \tag{6}$$

where w is the window size and l number of skipped lower frequency values. The texture response T_i of a region i is calculated as the mean LU transform value of a sampled set of points (N_i) inside the bounding box of the region.

$$T_i = \frac{1}{|N_i|} \sum_{p \in N_i} \Omega_p(l, w) \qquad T_i > t \tag{7}$$

Figure 5 shows the output of the of LU transform on an example image. In all the filters above, the thresholds were determined empirically and fixed in all our experiments to: $a_{\min} = 0.1$, $a_{\max} = 5$, $c = 1.4$, $e = 40$ and $t = 1.9$.

Fig. 5. LU transform output on an example image

3.3 Perceptual Text Grouping

After the image segmentation step, which produces a set of candidate text regions (usually representing isolated letters), a perceptual grouping step is performed to join them into candidate words and phrases. First, a planar Delaunay graph is constructed joining the centre of gravity of every text region. Each vertex of the graph represents a single text region, while the edges represent proximity relationships. Next, each edge e is then filtered using a sequence of tests.

Edge Angle – The first test looks at the angle between edges and the horizontal axis ($\alpha(e)$), such that,

$$-45^\circ < \alpha(e) < 45^\circ \tag{8}$$

This is a strong limitation but the majority of text is horizontal or with a slight slope. The angle of the text is also limited by the capabilities of the OCR engine used, as for now we are not performing any perspective correction.

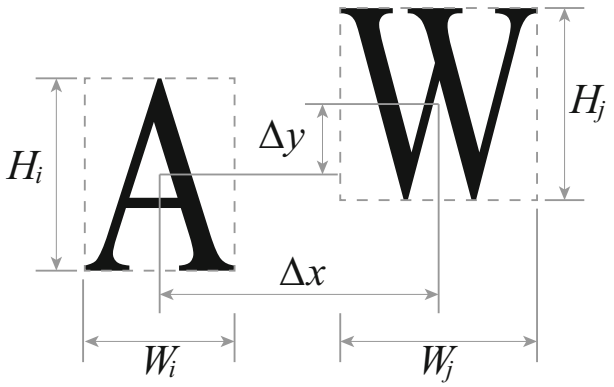


Fig. 6. Variables used for text grouping

Relative Position of Adjacent Tegions – The following criteria were inspired by the work of Ezaki et al. [6]. Two letters appearing on the same text line are usually close together. In this test we limit the allowed distance, relative to their respective sizes.

$$\Delta x < r_x \max(H_i, H_j) \qquad \Delta y < r_y \max(W_i, W_j) \tag{9}$$

where (H_i, W_i) and (H_j, W_j) are the bounding box dimensions of both regions, and $(\Delta x, \Delta y)$ represents the distance between the centres of both regions' bounding boxes (Figure 6). (r_x, r_y) are the *proximity coefficients*.

Size of Adjacent Regions – Similarly to the last test, two letters laying on the same line are assumed to have a similar size. This test limits the variance of adjacent region sizes.

$$\frac{|H_i - H_j|}{|H_i + H_j|} < r_h \qquad \frac{|W_i - W_j|}{|W_i + W_j|} < r_w \tag{10}$$

where (r_h, r_w) are the *size coefficients*, also determined experimentally.

After the edge filtering stage every remaining connected subgraph represents a text group. Text groups are tracked on consecutive frames and sent to the OCR engine for recognition.

4 Results

To facilitate comparative analysis, we measure performance on single image text detection on the ICDAR 2003 text localisation competition 'TrialTrain' dataset [10]. The same definitions for *precision* and *recall* were used as defined by the competition. However, given that our algorithm detects whole sentences instead of isolated words, we joined the bounding boxes of the ICDAR database words into sentences, to be able to make fair evaluations. This is the same approach that Pan et al. [19] employed.

The performance result[4] of the proposed method is shown in Table 1 along with the reported detection results from ICDAR 2003 and ICDAR 2005 text location competitions (average, and winning entries), as well as our previous method [15] and three other recent and state-of-the-art algorithms [19,17], and [5].

Fig. 7. Example results for some of the ICDAR 2003 database images

The proposed method shows a recall value of 0.67, close to the currently best performing algorithms, e.g 0.71 of [19], while not managing to obtain comparable precision performance. This means that our algorithm overestimates the number of detected regions, but indeed, it is not missing many real text locations. The lower precision rate can be compensated by the OCR engine discarding the unrecognisable regions. The text tracking step can also help in discarding the

[4] All results were obtained using an Intel Core 2 Duo T9300 CPU.

false positives as these non-text regions are unstable, while text regions are more consistently detected across several frames. In fact, by performing registration and super-resolution on tracked text regions [12], recognition accuracy can be increased. This is however beyond the scope of this paper and forms part of our future work. Some example results are shown in Fig. 7.

Table 1. Text detection performance on the ICDAR 2003 database

Text localization	prec.	recall	f	time (s)
Ashida (2003 winner) [10]	0.55	0.46	0.50	8.5
ICDAR 2003 average [10]	0.32	0.32	0.31	5.3
Hinnerk Becker (2005 winner) [11]	0.62	0.67	0.64	14.4
ICDAR 2005 average [11]	0.39	0.46	0.39	4.25
Merino and Mirmehdi [15]	0.44	0.68	0.48	0.1
Neumann and Matas [17]	0.59	0.55	0.57	N/A
Epshtein et al. [5]	0.73	0.60	0.66	0.94
Pan et al. [19]	0.67	0.71	0.69	2.43
Proposed method	0.51	0.67	0.55	0.2

One key advantage of our implementation is its simplicity and speed, which makes it feasible for real-time applications, including those involving text tracking. On the ICDAR database, it takes an average of 156 ms per image, but this is not representative for a real-time video text processor as every ICDAR database image has a different size and they are mostly high resolution still images. For video sequences we are able to process 14fps on 640×480 images and 9fps on 800×600 images (see Table 2).

Table 2. Time consumptions of different stages of the text locator

	MSER	Filtering	**total**	
ICDAR database	134 ms	16 ms	156 ms	
640×480 video	49 ms	10 ms	61 ms	14 fps
800×600 video	74 ms	15 ms	95 ms	9 fps

5 Conclusion

We have reported a wearable text recognition tool that employs MSERs as the basis for real-time text detection. The proposed method refines our previous real time algorithm by exploiting hierarchical structure obtained from MSERs to yield more stable regions compared to the previous adaptive threshold method. It outperforms other published approaches computationally while maintaining

similar text detection performance on the ICDAR dataset. In our future work, we plan to explore the introduction of a training stage for character recognition without reliance on third-party software, adding more cascading filters, and improving precision and recall results in general.

Acknowledgements. The Hat's construction was carried out at Bristol University based on a previous prototype and work carried out at University of La Laguna by Carlos Merino Gracia under the direction of José Luis González Mora.

Karel Lenc's contribution to this work was carried out at Bristol University as an ERASMUS student.

The Spanish Ministerio de Industria, Turismo y Comercio funds Carlos Merino Gracia through the European Regional Development Fund (project TSI-020100-2009-541).

References

1. Aoki, H., Schiele, B., Pentland, A.: Realtime personal positioning system for wearable computers. In: ISWC 1999, pp. 37–43. IEEE Computer Society, Washington, DC, USA (1999)
2. Chmiel, J., Stankiewicz, O., Switala, W., Tluczek, M., Jelonek, J.: Read IT project report: A portable text reading system for the blind people (2005)
3. Donoser, M., Bischof, H.: Efficient maximally stable extremal region (MSER) tracking. In: CVPR 2006, pp. 553–560 (2006)
4. Donoser, M., Arth, C., Bischof, H.: Detecting, Tracking and Recognizing License Plates. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part II. LNCS, vol. 4844, pp. 447–456. Springer, Heidelberg (2007)
5. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: CVPR 2010, pp. 2963–2970 (2010)
6. Ezaki, N., Kiyota, K., Minh, B., Bulacu, M., Schomaker, L.: Improved text-detection methods for a camera-based text reading system for blind persons. In: ICDAR 2005, pp. 257–261 (2005)
7. Hedgpeth, T., Black, J.A., Panchanathan, S.: A demonstration of the iCARE portable reader. In: ASSETS 2006, pp. 279–280 (2006)
8. Kurzweil, R.: The age of spiritual machines: when computers exceed human intelligence. Viking Press (1998)
9. Liang, J., Doermann, D., Li, H.: Camera-based analysis of text and documents: a survey. IJDAR, 84–104 (2005)
10. Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: ICDAR 2003 robust reading competitions. In: ICDAR 2003, pp. 682–687 (2003)
11. Lucas, S.: ICDAR 2005 text locating competition results. In: ICDAR 2005, pp. 80–84 (2005)
12. Mancas-Thillou, C., Mirmehdi, M.: Super-resolution text using the teager filter. In: CBDAR 2005, pp. 10–16 (2005)
13. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: BMVC 2002 (2002)
14. Mayol, W.W., Tordoff, B.J., Murray, D.W.: Wearable visual robots. Personal and Ubiquitous Computing 6, 37–48 (2002)

15. Merino, C., Mirmehdi, M.: A framework towards realtime detection and tracking of text. In: CBDAR 2007, pp. 10–17 (2007)
16. Myers, G.K., Burns, B.: A robust method for tracking scene text in video imagery. In: CBDAR 2005 (2005)
17. Neumann, L., Matas, J.: A Method for Text Localization and Recognition in Real-World Images. In: Kimmel, R., Klette, R., Sugimoto, A. (eds.) ACCV 2010, Part III. LNCS, vol. 6494, pp. 770–783. Springer, Heidelberg (2011)
18. Nistér, D., Stewénius, H.: Linear Time Maximally Stable Extremal Regions. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 183–196. Springer, Heidelberg (2008)
19. Pan, Y.F., Hou, X., Liu, C.L.: Text localization in natural scene images based on conditional random field. In: ICDAR 2009, pp. 6–10 (2009)
20. Pan, Y.F., Hou, X., Liu, C.L.: A hybrid approach to detect and localize texts in natural scene images. TIP (2011)
21. Peters, J.P., Thillou, C., Ferreira, S.: Embedded reading device for blind people: a user-centred design. In: AIPR 2004, pp. 217–222 (2004)
22. Shi, X., Xu, Y.: A wearable translation robot. In: ICRA 2005 (2005)
23. Targhi, A.T., Hayman, E., Olof Eklundh, J.: Real-time texture detection using the LU-transform. In: CIMCV (2006)
24. Zhang, J., Kasturi, R.: Extraction of text objects in video documents: Recent progress. In: DAS 2008, pp. 5–17. IEEE Computer Society, Washington, DC, USA (2008)

Appendix D

Fast Perspective Recovery of Text in Natural Scenes

This appendix includes the full text for the following article. The article is part of the PhD by publication:

Title	Fast perspective recovery of text in natural scenes.
Authors	Carlos Merino-Gracia, Majid Mirmehdi, José Sigut and José L. González-Mora
Type	Journal
Journal	Image and Vision Computing
Year	2013
Volume	31
Number	10
Pages	714–724
ISSN	0262-8856
DOI	10.1016/j.imavis.2013.07.002
URL	http://www.sciencedirect.com/science/article/pii/S0262885613001066

Image and Vision Computing 31 (2013) 714–724

Contents lists available at SciVerse ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

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

ARTICLE INFO

Article history:
Received 18 July 2012
Received in revised form 13 June 2013
Accepted 18 July 2013
Available online 26 July 2013

Keywords:
Scene text extraction
Perspective recovery
Homography rectification

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.

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 recognize 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 ground truth 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 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.

☆ This paper has been recommended for acceptance by Cheng-Lin Liu.

☆☆ 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).

* Corresponding author at: Neurochemistry and Neuroimaging Laboratory, University of La Laguna, Spain. Tel.: +34 922 319363.
E-mail addresses: cmerino@ull.es (C. Merino-Gracia), majid@cs.bris.ac.uk (M. Mirmehdi), sigut@isaatc.ull.es (J. Sigut), jlgonzal@ull.es (J.L. González-Mora).

0262-8856/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.imavis.2013.07.002

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 modeled 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:

$$\mathbf{p}' = \mathbf{H}\mathbf{p}, \tag{1}$$

where $\mathbf{p} = [x\ y\ 1]^\top$ and $\mathbf{p}' = [cx'\ cy'\ c]^\top$ 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 center. 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.

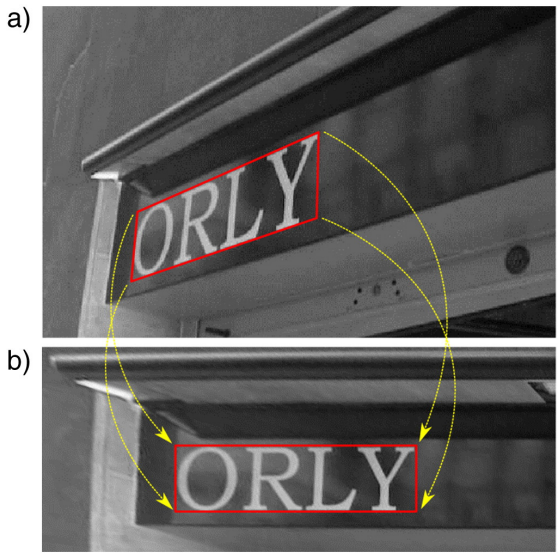


Fig. 1. *A projective transformation of text.* A rectangle enclosing the text is seen as a quadrilateral in the *source image* (a) and is mapped into a rectangle in the *target image* (b).

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 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,

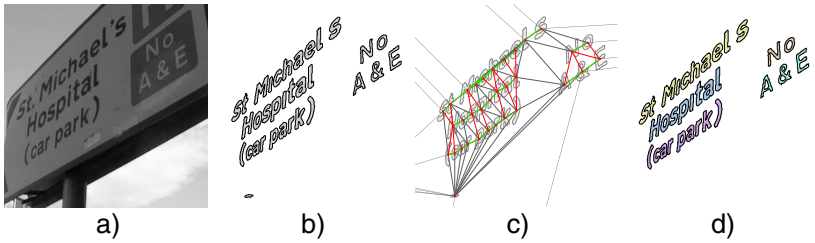


Fig. 2. The result of the segmentation and grouping steps: (a) the original image, (b) the segmented components, (c) the association graph (gray edges were removed during saliency filtering and red edges were removed during histogram filtering; the green edges represent the segmented text lines), and (d) the grouped text lines. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

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 center 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 recognizing *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 center 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 recognizing characters with perspective distortion directly. This technique was aimed at recognizing 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 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

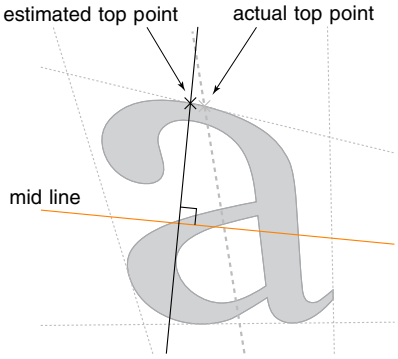


Fig. 3. Top point estimation — on severely perspective distorted characters the estimated top point and the actual top point might not correspond.

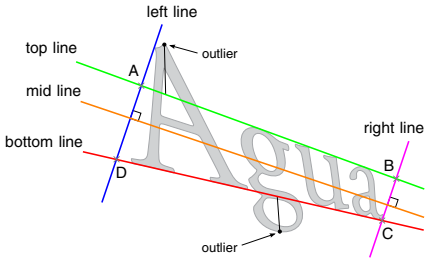


Fig. 4. Top, mid, bottom, left and right lines along with the formed quadrilateral. The outliers of the line estimation are also illustrated here.

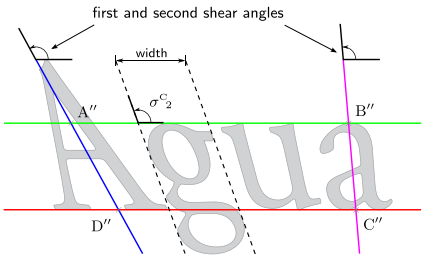


Fig. 6. Quadrilateral formed after computing the two shear angles. Additionally, the *upright shear angle* (σ^C) for the second character is shown.

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]. Fig. 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.

3.2.1. Saliency filtering

First, a Delaunay triangulation [25] joining the center points of every CC is performed, with the center points being the center of mass of each region. The Delaunay triangulation enables us to efficiently construct a neighbor relationship graph between all the components. Fig. 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.

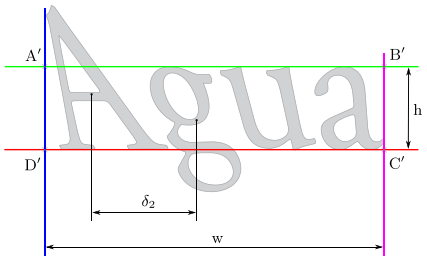


Fig. 5. Image partially rectified according to the top, bottom, left and right lines. The displacement (δ_2) for the second character is also shown.

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.

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

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.

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

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:

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

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 gray.

3.2.2. Histogram filtering

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

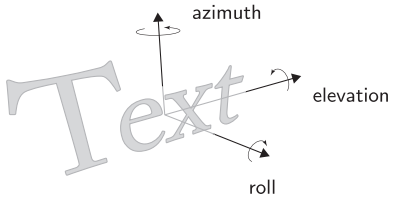


Fig. 7. Axes for the rotations applied to text in our experiments.

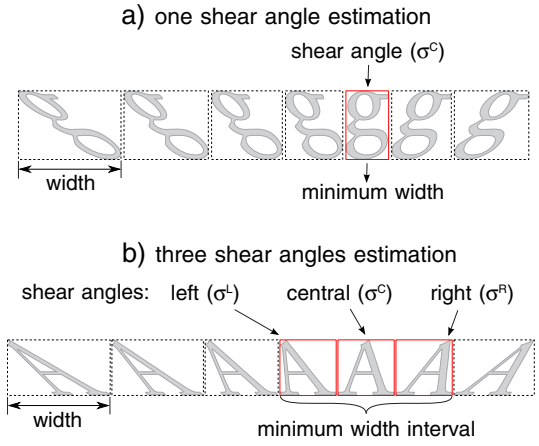


Fig. 8. Shear estimation for one character.

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 C_i, $i = 1,...N$, 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.

3.3.1. Parallel rectification

A line is fitted to the center point of every character in the text line (the center 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' center 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, labeled clockwise starting with the intersection of the left and top lines, as in the example shown in Fig. 4.

A straightforward homography $\mathbf{H}_p$ 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 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:

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

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*.

3.3.2. 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

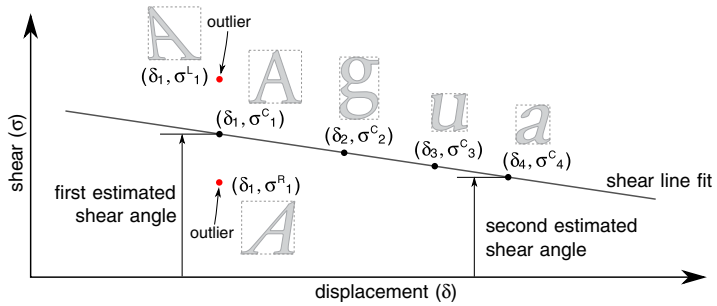


Fig. 9. Shear angle estimation — from the alternative shear candidates of the first letter ($\sigma^L{}_0, \sigma^C{}_0$ and $\sigma^R{}_0$) the wrong ones (in red) are discarded as outliers by the RANSAC line fitting. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

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' center points are ordered along the x-axis in the parallel image. The horizontal distance of each character's center 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 letter '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 $\sigma^C = (\sigma^L + \sigma^R)/2$. 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: (δ_i, ${\sigma^C}_i$), (δ_i, ${\sigma^L}_i$) and (δ_i, ${\sigma^R}_i$), $i = 1,...N$. 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. ${\sigma^C}_2$) 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 defined on both sides of the text line. They pass through the center of the left-most and right-most characters respectively and form 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″B″, as shown in the example in Fig. 6). A homography $\mathbf{H}_s$ 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:

$$\mathbf{H} = \mathbf{H}_s\mathbf{H}_p. \tag{6}$$

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

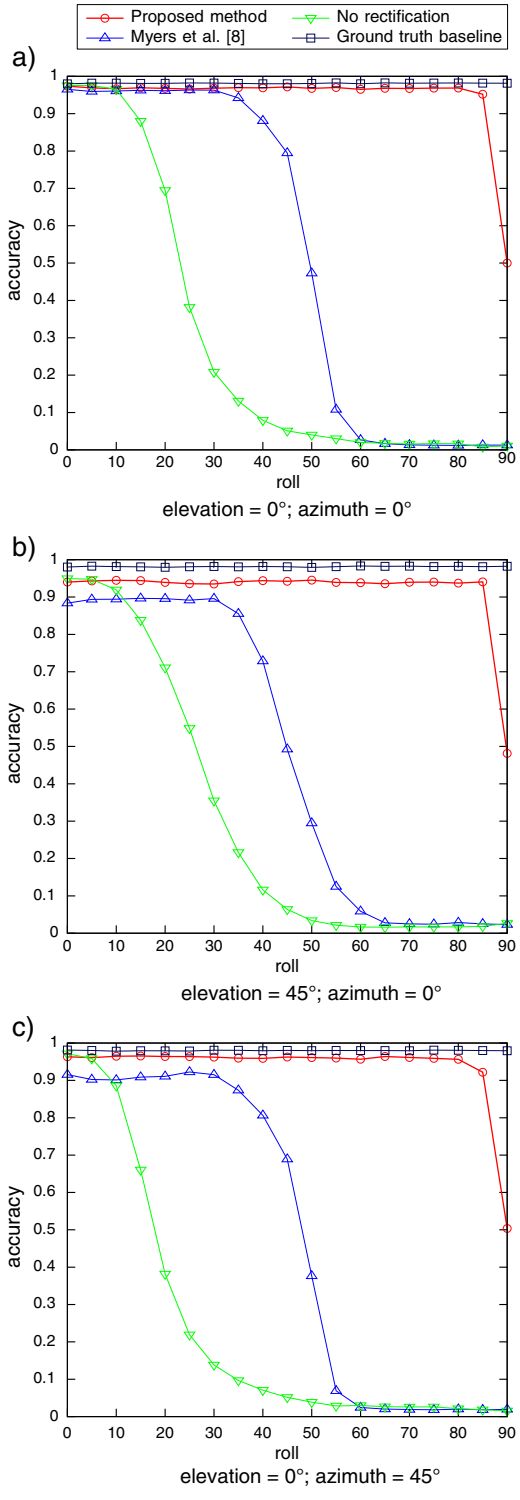


Fig. 10. The effect of roll on recognition accuracy.

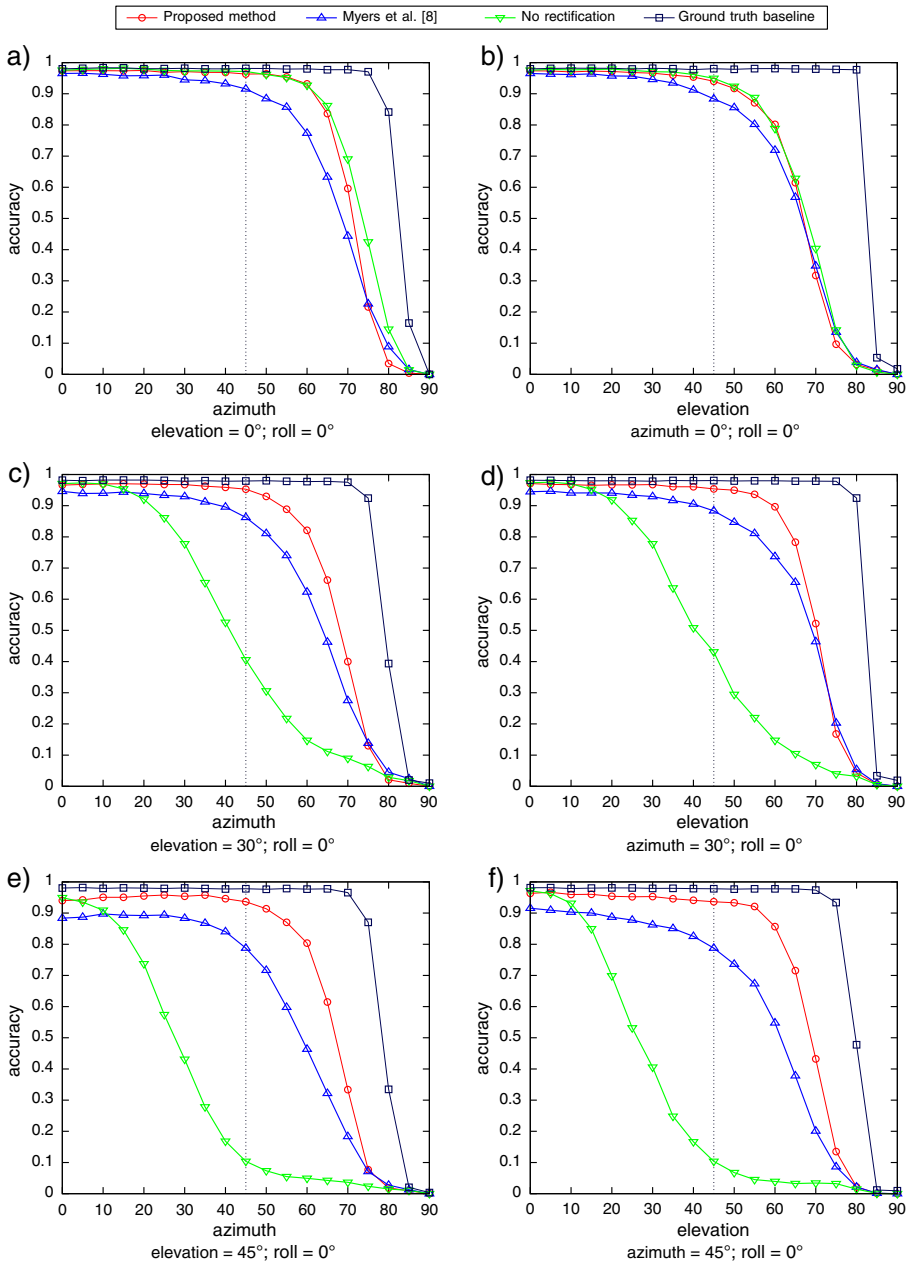


Fig. 11. The effect of azimuth and elevation on recognition accuracy.

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 localization mistakes that would arise from using real-world images, and so we obtain an

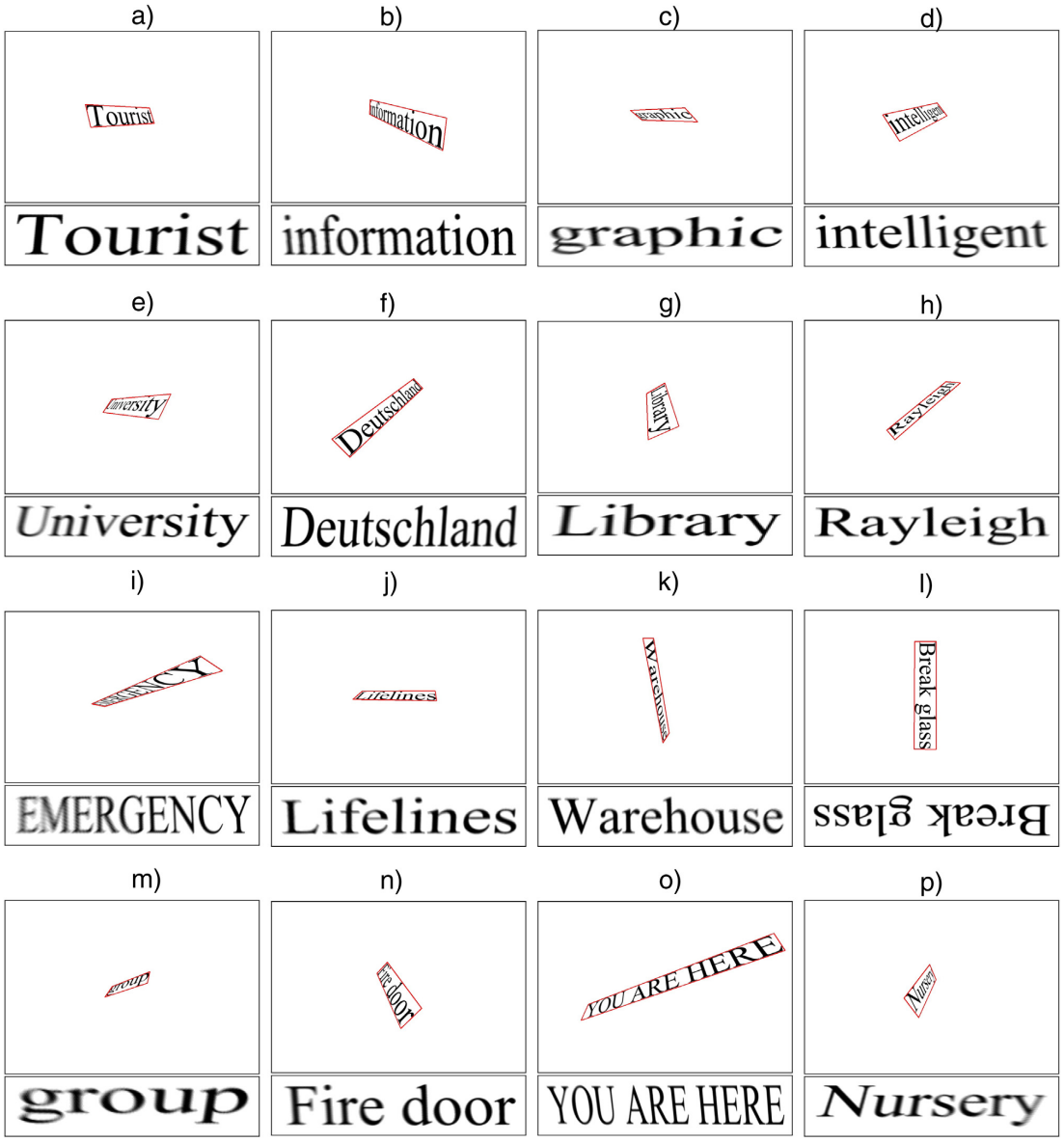


Fig. 12. A selection of the synthetic images used in the experiments, along with their *estimated* orientation (red box) and corresponding rectified image. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

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 [8] method 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,

[1] http://code.google.com/p/tesseract-ocr/.

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.

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

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. Fig. 12i, m or p, 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 Fig. 10b and c 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. Fig. 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°.

Fig. 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° — Fig. 11a, c and e respectively) and likewise, the right column displays the variation of elevation for fixed values of azimuth (0°, 30° and 45° — Fig. 11b, d and f 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 eta 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 (Fig. 11d and f) 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.

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 labeled 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. Fig. 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 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 Eq. (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.

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 angle 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, i, o, 14a, n, and o, the 'T' in Figs. 12a, 14g, and o, or the 'W' in Figs. 12k, 14e, and o). In

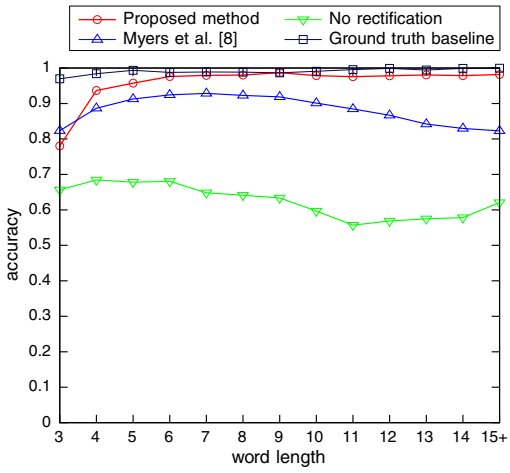


Fig. 13. The effect of word length on recognition accuracy.

Fig. 14. A selection of real world images with scene text, along with the text's estimated orientation (red box) and rectified image. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

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 Fig. 14b, f and m).

4.3. Speed

In our implementation, text extraction, including segmentation, grouping and perspective estimation, performed on an Intel Core

Table 1
Word length distribution in the synthetic text dataset.

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

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

i7-2600 processor, achieved real-time performance of 20 fps on 1280×720 video sequences. The orientation detection stage (as explained in Section 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 the 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 labeled as text can confuse the top/bottom line 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 recognizable 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, CBDAR'11, vol. 7139 of *LNCS*, Springer Berlin, Heidelberg, 2012, pp. 29–41.
[2] J. Gao, J. Yang, An adaptive algorithm for text detection from natural scenes, CVPR'01, vol. 2, II, 2001, pp. 84–89.
[3] C. Mancas-Thillou, S. Ferreira, J. Demeyer, C. Minetti, B. Gosselin, A multifunctional reading assistant for the visually impaired, JIVP 5 (2007) 1–11.
[4] I. Posner, P. Corke, P. Newman, Using text-spotting to query the world, IROS'10, 2010, pp. 3181–3186.
[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), ICDAR'11, 2011, pp. 1485–1490.
[7] A. Shahab, F. Shafait, A. Dengel, ICDAR 2011 robust reading competition challenge 2: reading text in scene images, ICDAR'11, 2011, pp. 1491–1496.
[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, second ed. Cambridge University Press, 2004.
[13] M. Pilu, Extraction of illusory linear clues in perspectively skewed documents, CVPR, 2001, pp. 363–368.
[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, ICCV Wshps, 2011, pp. 64–71.
[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, CBDAR'07, 2007, pp. 10–17.
[22] Y.-F. Pan, X. Hou, C.-L. Liu, Text localization in natural scene images based on conditional random field, ICDAR'09, 2009, pp. 6–10.
[23] B. Epshtein, E. Ofek, Y. Wexler, Detecting text in natural scenes with stroke width transform, CVPR, 2010, pp. 2963–2970.
[24] A. Targhi, E. Hayman, J. Eklundh, M. Shahshahani, The eigen-transform & applications, ACCV, I, 2006, pp. 70–79.
[25] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, second edn Springer-Verlag, 2000.
[26] G. Toussaint, Solving geometric problems with the rotating calipers, IEEE MELECON'83, 1983, pp. 10–17.
[27] S.M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, ICDAR 2003 robust reading competitions, ICDAR'03, 2003, pp. 682–687.
[28] K. Wang, B. Babenko, S. Belongie, End-to-end scene text recognition, ICCV'11, Barcelona, Spain, 2011, pp. 1457–1464.

Appendix E

Real-time Text Tracking in Natural Scenes

This appendix includes the full text for the following article. The article is part of the PhD by publication:

Title	Real-time text tracking in natural scenes.
Authors	Carlos Merino-Gracia and Majid Mirmehdi
Type	Journal
Journal	IET Computer Vision
Year	2014
Volume	8
Number	6
Pages	670–681
ISSN	1751-9632
DOI	10.1049/iet-cvi.2013.0217
URL	http://dx.doi.org/10.1049/iet-cvi.2013.0217

www.ietdl.org

Published in IET Computer Vision
Received on 17th August 2013
Revised on 24th March 2014
Accepted on 28th March 2014
doi: 10.1049/iet-cvi.2013.0217

ISSN 1751-9632

Real-time text tracking in natural scenes

Carlos Merino-Gracia[1,2], Majid Mirmehdi[2]

[1]*Neurochemistry and Neuroimaging Laboratory, University of La Laguna, La Laguna, Spain*
[2]*Visual Information Laboratory, University of Bristol, Bristol, UK*
E-mail: cmerino@ull.es

Abstract: The authors present a system that automatically detects, recognises and tracks text in natural scenes in real-time. The focus of the author's method is on large text found in outdoor environments, such as shop signs, street names, billboards and so on. Built on top of their previously developed techniques for scene text detection and orientation estimation, the main contribution of this work is to present a complete end-to-end scene text reading system based on text tracking. They propose to use a set of unscented Kalman filters to maintain each text region's identity and to continuously track the homography transformation of the text into a fronto-parallel view, thereby being resilient to erratic camera motion and wide baseline changes in orientation. The system is designed for continuous, unsupervised operation in a handheld or wearable system over long periods of time. It is completely automatic and features quick failure recovery and interactive text reading. It is also highly parallelised to maximise usage of available processing power and achieve real-time operation. They demonstrate the performance of the system on sequences recorded in outdoor scenarios.

1 Introduction

Accessing textual information in the environment is crucial in our daily lives and there is a clear need for technology that can automatically extract and process such text for the benefit of those who might have difficulty accessing it, for example, for assisting the blind, for translating for tourists and for devices that need to know, such as robots. Hence, in recent years, there has been a significant increase in scene text detection and recognition works, see for example [1–4]. The main focus of such works has been on text segmentation in single images, with increasingly better results as measured against the most widely used dataset [5]. However, current state-of-the-art scene text recognition methods, such as those above, and including those that operate on video images and in real-time, for example, [3], lack scene awareness. They treat their input as a succession of unrelated images, attempting to segment and recognise the text in them without taking advantage of the fact that the same text is invariably repeated across many consecutive frames in a video sequence.

Natural scene images pose significant challenges to text understanding, such as blurred or out of focus frames, uneven lighting, complex backgrounds or perspective and lens distortion. On the other hand, a continuous stream of frames provides a temporal redundancy that can help address some of these drawbacks. For example, a blurred image can be difficult or impossible to process on its own, but as part of a sequence, a blurred frame can be ignored as there are chances that other frames in the sequence are clear. In addition, while the estimation of the three-dimensional (3D) orientation of scene text from single images is certainly possible [6, 7], the apparent changes objects undergo in the scene as the camera moves can help reduce the uncertainty in orientation estimation.

The main contribution of the work here is a text reading prototype based on text tracking. Text tracking leverages the main difference between flat-bed scanner and camera-based document analysis (i.e. spatial resolution against temporal redundancy) and allows us (i) to maintain region identity across the sequence and (ii) to smooth the estimation of the region's parameters (position and 3D orientation) to reduce jitter. Both of these outcomes play a major role in facilitating scene awareness, in reduction of false positive segmentations and increase in recognition accuracy, and in better interactive communication of text in the environment to the user, for example, by managing the frequency of communicating text seen in the scene as an audio signal to a blind user. Text regions are segmented using an adaptive threshold-based technique [8], and their 3D orientation estimated by means of an efficient geometrical algorithm [7]. Then, each text region is independently tracked by an unscented Kalman filter (UKF). Our prototype operates in real-time and it is autonomous, that is, new trackers are created when new text entities are found and their identity is kept for as long they are in view; they are removed when the entities are no longer detected, given some resilience to brief occlusions. Trackers are automatically selected for optical character recognition (OCR) – a process carried out in parallel to tracking by an off-the-shelf OCR engine. This demonstrates the role tracking plays, that is, when a text recognition result is available, the system is able to relate it to the original text entity even if the camera has moved. Likewise, available recognition results are automatically selected for synthesising into audio (using text-to-speech) and are played back to the user.

This paper is organised as follows. Section 2 outlines related works. The proposed system is described in detail in Section 3 and in particular, Section 4 explains the text tracking mechanism. We discuss the quantitative and qualitative performance results of our system in Section 5. Finally, Section 6 concludes the paper.

2 Related work

There are very few works on text tracking, let alone natural scene text tracking. Initial works in this area focused on tracking graphically overlaid text in videos (e.g. in news or sports reports), such as [9–11], and here we will not deal with such works any further. In the area of scene text tracking, we are only aware of these works [8, 12–18]. The earliest work specifically dealing with scene text tracking is that of Myers and Burns [12], where an offline system was proposed to extract scene text at arbitrary orientations. They first tracked a series of feature points across the whole sequence, and then, with the assumption that all the points belonging to a certain text area lie on the same plane, they estimated the planar transformation of the points in multiple frames simultaneously to extract a fronto-parallel representation. As it stands, this technique is unsuitable for online real-time operation.

In [8], we presented a near real-time (15 fps) scene text tracking system-based on particle filtering (PF). The text was first segmented using an adaptive threshold-based technique. Segmented components were then grouped together using a saliency filter to form text entities (i.e. words or groups of words forming small sentences). Each text entity was assigned a PF tracker which maintained a set of features and a simple state (a 2D translation and an in-plane rotation). The particles' weights were computed as the number of matched features within a search area, where the identity of individual features was established using scale invariant feature transform (SIFT) [19] descriptors. A key aspect of this method was the use of just a few high quality features for tracking – in this case, segmented characters. This required a full text segmentation stage per frame (and thus demanded a very fast text segmentation algorithm), but the advantage was that the trackers were more resilient to big changes in orientation, occlusions and illumination changes. This early work, although very useful as a proof of concept, was found to have certain drawbacks where some performance improvement was necessary. For instance, over 80% of the frame processing rate was associated with feature matching, that is, SIFT, which is a computationally expensive operator. In addition, the number of SIFT descriptors produced by each feature (i.e. characters) was rather low, and limited the ability to discriminate between measurements. SIFT is affine invariant but not perspective invariant, and no estimation of the 3D spatial orientation of the text in the scene was performed so the whole system was sensitive to wide baseline changes. This is also related to the simple state model used, namely just a 2D translation and in-plane rotation, that traded accuracy and robustness in favour of low computational complexity and hence better frame rate.

Particle filtering was also used by Tanaka and Goto [15, 16] and by Minetto *et al.* [18] for text tracking. The former works described a wearable system for the blind where text was detected using discrete cosine transform (DCT)-based features (on prior works by Goto and his co-workers [13, 14], a simple tracking system was described based on block matching between frames). Tracking was performed by generating particles on candidate text regions in new frames, and they were weighted according to a similarity function between the regions based on cumulative histograms. No perspective correction was performed, and only region identity and limited 2D motion was maintained by this method.

In Minetto *et al.* [18], candidate text regions were initially segmented using a morphological operator applied at different scales, then classified by means of a support vector machine, and grouped together based on their relative distances and sizes. For each text region, particles were propagated in subsequent frames using a first-order motion model. Particles' weights were proportional to a similarity coefficient between the histogram of oriented gradients descriptors of the respective image regions.

A different approach was used by Na and Wen [17] by tracking text directly using SIFT. A global motion between frames, modelled as a similarity transformation [20] was computed by minimising the least squares distance between the SIFT feature matches. In their work, the authors did not specify how text regions were segmented. Furthermore, the computational complexity of extracting SIFT descriptors from every frame precludes the real-time use of this technique, and the simple motion model limits the usefulness of the method when applied to outdoor hand-held camera scenarios. SIFT was also used in [21] to track and align text regions appearing in a sequence of frames. An integration of the aligned text probability maps was then used to improve OCR accuracy. Their algorithm looks for text in adjacent frames either forwards or backwards, necessarily making it an offline process and unsuitable for real-time operation. In addition, the method requires a manual selection of the initial text bounding box.

In the work by Mosleh *et al.* [22], text tracking is used to separate overlaid text from scene text in order to automatically remove the former (but not the latter) from video sequences. A CAMSHIFT algorithm is used to infer the text motion, assuming a few constraints on text movement (i.e. movement along the vertical or horizontal axis). Optical flow is then used to isolate the movement of these regions from that of the background. Other recent text detection works of note are [23–25], in which the focus is not only on segmenting the text but also recognising it. However, these approaches still lack the context and temporal redundancy awareness that tracking provides. To the authors' knowledge, no other work exists on scene text tracking which shows the scope, aim and performance of the methodology proposed here.

3 Preparing to track: text segmentation and perspective recovery

The proposed end-to-end real-time text reading system, including its tracking and other processing stages, is illustrated in Fig. 1. Text tracking is performed on high level features (i.e. perspective corrected characters). This provides several advantages over low-level tracking of feature points, such as increased resiliency against orientation changes and occlusions. However, the tracking needs to be preceded by quick and efficient techniques for text segmentation and perspective estimation, as these operations have to be performed on every frame. Here, we build on our previous work for real-time text detection, grouping and perspective rectification [7, 8, 26] and for the

IET Comput. Vis., pp. 1–12
doi: 10.1049/iet-cvi.2013.0217

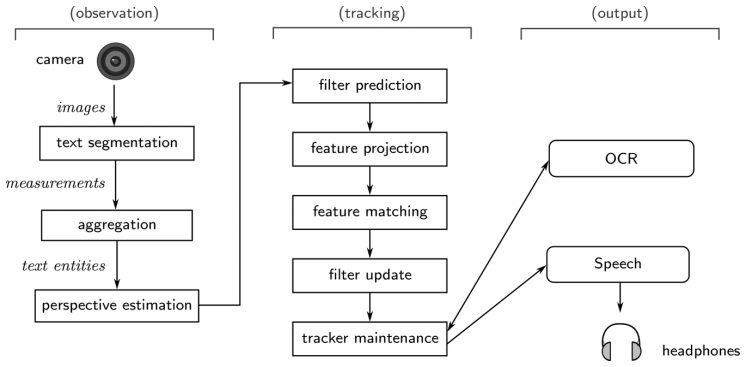


Fig. 1 *Schematic of the proposed end-to-end real-time text reading system*

sake of completeness, these fundamental stages are first briefly summarised in Sections 3.1–3.3. Then, the main focus of this work, that is, our text tracking mechanism, is explained in detail in Section 4.

3.1 Text detection

Each input image is segmented to obtain a set of candidate text regions, containing possibly none, or one, or more characters. Later, these regions will be used as measurements for the tracking filter, as well as serving as the building blocks for text line aggregation and tracker creation.

A brief outline of our segmentation algorithm, first presented in [8], is as follows. Adaptive thresholding is applied to binarise the input image and retrieve a set of connected component (CC) regions. 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 in natural scene images with text, structural elements (such as sign borders, posters frames etc.) can be discarded purely based on their hierarchical relationships with other text regions. In addition, the tree filtering approach we proposed in [8] allows for the segmentation of dark and light text in one pass only. The CCs are then pruned 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 [27]. Fig. 2*a* shows an example image and Fig. 2*b* illustrates the corresponding CCs (i.e. candidate text regions) detected at this stage.

3.2 Text aggregation

For the purpose of text recognition by (off-the-shelf) OCR, we need to group the CC regions together in order to form text entities (as we previously performed in [8]) for tracking and to be able to extract common clues for perspective estimation. The grouping is performed by first determining which CC regions are closely associated by evaluating a visual saliency measure between each pair of regions, and then by searching for dominant orientations to separate

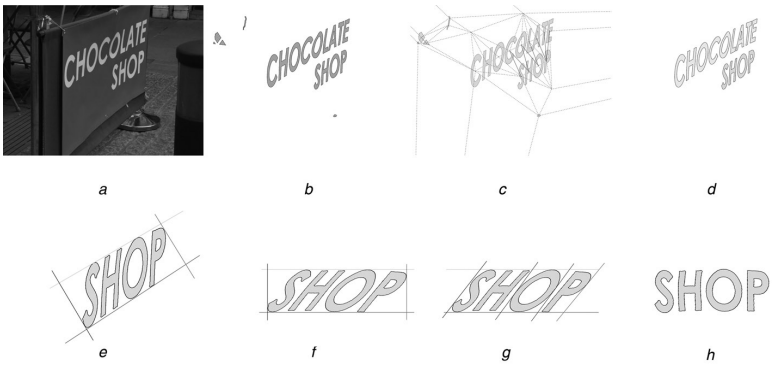


Fig. 2 *Text segmentation and perspective estimation for*

a Example of original image
First, for the segmentation and aggregation stages
b Segmented components
c Association graph (dashed edges were removed during saliency filtering and thin edges were removed during histogram filtering; the thick edges represent the segmented text lines)
d Grouped text lines
Then, for the perspective estimation step
e Top and bottom lines estimation
f Parallel image
g Shear estimation
Finally
h Full perspective rectification

independent lines of text. First, a Delaunay triangulation [28] to join 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. Fig. 2*c* 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 [29]. Those edges with a low saliency value are removed from the graph. In Fig. 2*c*, edges removed during the saliency filtering are represented in grey.

After the saliency filtering, every remaining connected subgraph is a candidate text group, each of them possibly containing one or more lines of text. A histogram of angle distribution of the graph edges is constructed to find each group's dominant orientation, which is chosen to be the histogram bin with highest element count. The edges of the graph that do not belong to this dominant orientation are removed, after which the original subgraph may be split into smaller subgraphs separating the individual text lines. In Fig. 2*c*, filtered edges at this stage are represented in red, and the remaining connected subgraphs are represented in green. Fig. 2*d* shows the result of the text segmentation and grouping, in which each segmented text line is drawn in a different colour. Every remaining connected subgraph contains only one text line and is called a text entity. Text entities are the basis of new tracker creation (as explained later in Section 4.1).

3.3 Perspective estimation

At this stage of text recovery, we estimate a 3×3 homography matrix transformation of a text entity into a fronto-parallel representation [7]. Assuming a pinhole camera model, the homography is the transformation that allows the modelling of all possible orientation changes the text can undergo in an image without the need to have calibrated cameras or an explicit 3D representation. The ability to quickly estimate the text orientation makes high-level perspective-aware tracking possible, as well as the extraction of perspective covariant feature descriptors.

The orientation detection is performed in two steps: parallel rectification and shear estimation. At first we compute the farthest character points on each side of the main axis of the text line, which are used to fit a top and a bottom line (Fig. 2*e*). As both lines are assumed to be parallel in the real world, they are used to rotate and partially rectify the text entity leaving only a remaining shearing effect (Fig. 2*f*).

Then, we estimate a shear value for every character by minimising the distance between two parallel lines that rotate around antipodal vertices of the character's contour (this is a variation of the Rotating Calipers paradigm [30]). With a shear value for each character, a linear shear variation across the whole line can be obtained and used to compute the full rectifying homography of the text entity, as illustrated in Fig. 2*g*. Fig. 2*h* shows the rectified image.

4 Proposed text tracking

We now present our proposed text tracking approach. Once a text entity is identified, a tracker is created to follow the text region from frame to frame while it is in camera view. The detailed process of tracker creation and removal is explained later in Section 4.1. For now, for ease of exposition, we assume that a set of trackers already exists and properly initialised to follow a corresponding set of text entities in the scene.

A tracker is characterised by a set of tracked features $\boldsymbol{z}_i$ and a dynamic state $\boldsymbol{x}_k$, which is updated by a predictive filter. The features $\boldsymbol{z}_i$ correspond to the individual characters in the text line and are used as the anchor points to be matched against image measurements during the observation stage of the filter. They are stored in a fronto-parallel representation, in a coordinate frame referred to as 'tracker coordinates' (see Fig. 3). Each feature is defined as

$$\boldsymbol{z}_i = \begin{bmatrix} x & y & w & h \end{bmatrix}^{\mathrm{T}}, \quad i = 1, \ldots, M \tag{1}$$

where (x, y) is a feature's centre point, (w, h) are the dimensions of the feature's bounding box and M is the number of features. In addition, each feature keeps a perspective corrected image patch used during feature matching (as seen in Fig. 3).

In [8], we used PFs [31] since they model a non-linear system, such as our text tracking problem, well. However, in this work we choose the unscented Kalman filter (UKF) [32] for tracking because it provides the uncertainty of the system's state estimation via a Gaussian probability distribution, and it is also more efficient, that is, to achieve the same accuracy as the PF, it needs to use substantially fewer sampling points. The UKF is derivative free and employs a deterministic sampling approach (the unscented transform, UT) to propagate the density function across non-linear state changes. The UT captures the non-linearities better than alternatives such as the extended Kalman filter and is easier to implement.

We represent the filter state by a homography transformation $\boldsymbol{H}$ mapping the fronto-parallel view of the tracked features in tracker coordinates to image coordinates (Fig. 3). The homography can be characterised by a vector $\boldsymbol{h}$

$$\boldsymbol{h} = \begin{bmatrix} t_x & t_y & \theta & s_x & s_y & \sigma & l_x & l_y \end{bmatrix}^{\mathrm{T}} \tag{2}$$

where (t_x, t_y) defines a translation, θ is an in-plane rotation angle, (s_x, s_y) is an anisotropic scale, σ is a shearing and (l_x, l_y) is a foreshortening around both axes. Given the homography, there is a closed form unique solution for all

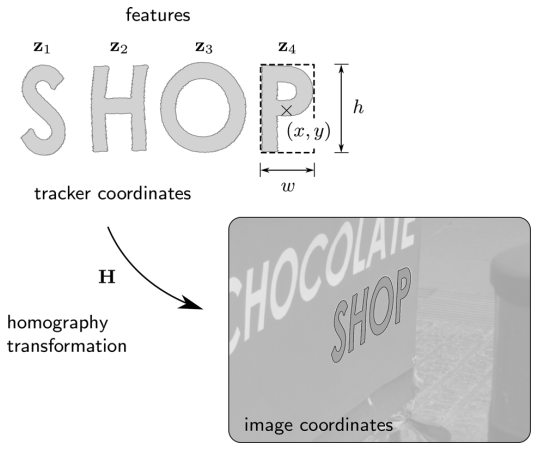


Fig. 3 *Tracker representation, which keeps a set of features $\boldsymbol{z}_i$ and the state of the UKF that produces the homography $\boldsymbol{H}$*

IET Comput. Vis., pp. 1–12
doi: 10.1049/iet-cvi.2013.0217

the parameters [20] (refer to the Appendix for the formulation of this solution) which we name, along with its inverse, the homography (de)composition function $\mathcal{H}$

$$\boldsymbol{H} = \mathcal{H}(\boldsymbol{h}), \quad \boldsymbol{h} = \mathcal{H}^{-1}(\boldsymbol{H}) \tag{3}$$

Although both representations are mathematically equivalent, with this decomposition the filter deals directly with the underlying parameters that define the transformation, enabling the direct estimation of the uncertainty in each parameter via the covariance matrices. The dynamic model also benefits from this representation as we can define velocity vectors that affect only the translation or rotation parameters of the transformation. For the rest of this section, when we refer to the homography $\boldsymbol{H}$, an implicit conversion will be assumed from the parameters vector $\boldsymbol{h}$ to the homography using $\mathcal{H}(\boldsymbol{h})$. The only moment in which the inverse operation $\mathcal{H}^{-1}(\boldsymbol{H})$ is needed is for tracker creation, as explained in Section 4.1.

For the prediction stage of the filter, we use a constant velocity model, where we only consider in-plane translational and angular velocities. We define the velocity vector $\boldsymbol{v}$ as

$$\boldsymbol{v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}^{\mathrm{T}} \tag{4}$$

and the state vector of the filter at frame k as a stacking of the homography parameters and velocity vectors

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{h} \\ \boldsymbol{v} \end{bmatrix} \tag{5}$$

The new state prediction is then

$$\hat{\boldsymbol{x}}_{k+1} = \begin{bmatrix} \boldsymbol{h} + \overset{\circ}{\boldsymbol{v}}\Delta t \\ \boldsymbol{v} \end{bmatrix} \tag{6}$$

where $\overset{\circ}{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{v}^{\mathrm{T}} & 0^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ is the velocity vector padded with zeros to the length of $\boldsymbol{h}$ and Δt is the elapsed time since the last frame.

The measurement function maps the tracked features to observable characteristics in the image (called measurements) using the filter state. However, as each tracker represents a line of text, the centre points of all the characters are roughly aligned. If the centre points were the only points of our measurement function, there would be a great deal of uncertainty for rotations around the horizontal axis (i.e. elevation – see Fig. 4 as an example). Since we have a good estimate of the text orientation, and we know that all the points of a text line lie on a plane, our observation model includes five points per tracked feature $\boldsymbol{z}_i$: the centre point $\hat{\boldsymbol{c}}_0$ and the four corner points $\hat{\boldsymbol{c}}_j$, $j = 1, \ldots, 4$ of the feature's bounding box (as shown in Fig. 5)

$$\hat{\boldsymbol{z}}_i = \begin{bmatrix} \hat{\boldsymbol{c}}_0^{\mathrm{T}} & \hat{\boldsymbol{c}}_1^{\mathrm{T}} & \hat{\boldsymbol{c}}_2^{\mathrm{T}} & \hat{\boldsymbol{c}}_3^{\mathrm{T}} & \hat{\boldsymbol{c}}_4^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \quad i = 1, \ldots, M \tag{7}$$

These are converted from tracker coordinates using the predicted state homography. For example, $\hat{\boldsymbol{c}}_0$ is computed as the transformation of (x, y) to image coordinates, $\hat{\boldsymbol{c}}_1$ as the transformation of $(x - w/2, y - h/2)$, and likewise for $\boldsymbol{c}_2$ to $\boldsymbol{c}_4$.

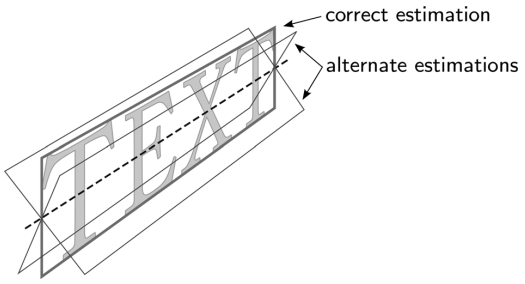


Fig. 4 *Homography estimation ambiguity if only the centre points of each character are considered*

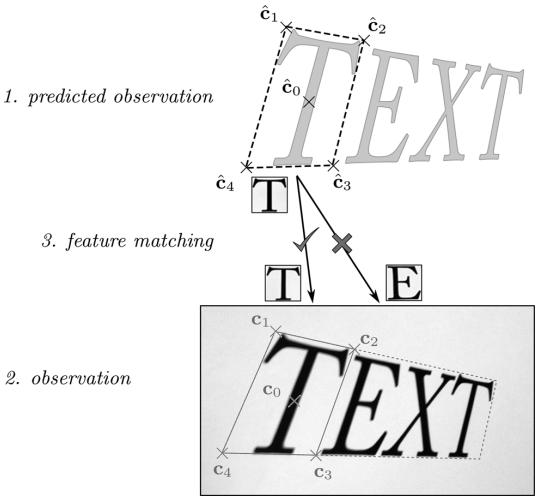


Fig. 5 *Observation model*

First a new location for the tracker features is predicted
Here the five predicted observation points $\hat{\boldsymbol{c}}_i$ for the first feature are represented in the upper part
Then, from the segmentation and perspective orientation stage, the actual measurement points $\boldsymbol{c}_i$ are obtained
These are represented in the lower part
Finally, the candidate measurements are matched using perspective corrected image patches which are also shown

The predicted observation is then a combination of all of the individual feature mappings

$$\hat{\boldsymbol{y}}_k = \begin{bmatrix} \hat{\boldsymbol{z}}_0^{\mathrm{T}} & \cdots & \hat{\boldsymbol{z}}_M^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{8}$$

Finally, as the last part of the observation model, the tracked features need to be matched against the segmented text regions or candidate measurements. To discriminate between the candidates, each feature keeps an image patch, normalised to 50×50 pixels. It is a perspective corrected image patch, extracted using the four corner points of the feature from the frame in which the tracker was created. Matching is performed using normalised cross correlation (NCC) and only on measurements within a certain search radius around the predicted feature position. The search radius is obtained from the filter's state covariance matrix, as it represents the uncertainty in the new state's prediction. After matching, each feature has one measurement candidate. As with the observation function, each

measurement $\boldsymbol{m}_i$ is defined by the centre point of the region and its four corner points

$$\boldsymbol{m}_i = \begin{bmatrix} \boldsymbol{c}_0^{\mathrm{T}} & \boldsymbol{c}_1^{\mathrm{T}} & \boldsymbol{c}_2^{\mathrm{T}} & \boldsymbol{c}_3^{\mathrm{T}} & \boldsymbol{c}_4^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \quad i = 1, \ldots, M \tag{9}$$

where $\boldsymbol{c}_j, j = 0, \ldots, 4$, are the mapped centre and corner points, and are obtained from the orientation estimation stage. Hence, the observation used by the UKF is

$$\boldsymbol{y}_k = \begin{bmatrix} \boldsymbol{m}_0^{\mathrm{T}} & \cdots & \boldsymbol{m}_M^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{10}$$

In Fig. 5, the observation model is illustrated: the predicted observation, the actual observation and the feature matching.

4.1 Tracker maintenance

Tracker maintenance refers to the set of mechanisms in which new trackers are created, new features are added to existing trackers, and bad trackers are removed. This allows the automatic continuous operation of the system.

At first we need to correlate the text entities produced in the text association stage to the current set of trackers. The text association stage groups measurements as belonging to the same text entity and the tracking stage may associate features to some of the measurements after matching.

By correlating tracked features to measurements and then to text entities, several possibilities arise: (i) a tracker has matched all the measurements belonging to a text entity – this is the perfect tracking case and no further action is needed (Fig. 6*a*); (ii) no tracker has matched any of the measurements of a given text entity – the entity is then a candidate for tracker creation (Fig. 6*b*); and (iii) a tracker has matched some of the measurements inside an entity – the remaining (unmatched) measurements are candidates for feature addition to that tracker (Fig. 6*c*). In addition, when a tracker matches most or all of its features it is considered a good track. Likewise, if a tracker did not match any of its features (or only matched a low fraction of them), it is considered a bad track or a mistrack. After a certain number of frames being a bad track, a tracker is removed. These operations are further explained in the following.

4.1.1 Tracker creation: When a text entity does not have any tracker matching any of its features, it is considered an untracked entity, thus requiring a tracker to be created for it. Tracker creation proceeds as follows: the perspective estimation stage returns a homography transformation $\boldsymbol{H}'$ of the measurements in the group to a fronto-parallel representation. All the measurements are converted into features in the new tracker by applying this homography transformation and then obtaining the centre point and dimensions of each text region in the fronto-parallel view. Then, the filter state is initialised as

$$\boldsymbol{x}_0 = \begin{bmatrix} \boldsymbol{h}_0^{\mathrm{T}} & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} \tag{11}$$

with $\boldsymbol{h}_0 = \mathcal{H}^{-1}(\boldsymbol{H}_0)$ and $\boldsymbol{H}_0 = (\boldsymbol{H}')^{-1}$ being the initial homography estimation of the transformation between the fronto-parallel representation to image coordinates.

On creation, a tracker is marked as unstable. This means that it will not be considered for feature addition, for recognition or transformation to speech, and it will not be shown as a segmented region. It is only considered stable after it is tracked continuously for a number of frames – in our case this was arbitrarily set to ten. As a text region is consistently segmented in a sequence of frames, as opposed to noisy regions, this process cleans most of the text segmentation false positives.

4.1.2 Feature addition: When a stable tracker matches some of the measurements inside a text entity, the remaining unmatched measurements are assumed to belong to the same entity. Hence, they are added to the tracker as new features. The corner points of the created feature are mapped to the tracker coordinates using the tracker's state homography, and the observation vector length is increased accordingly.

4.1.3 Tracker removal: When a tracker has been regarded a bad track for a few frames because none or too few of its features are matched, the tracker is removed. There is no long term registry of old trackers. If a tracker is removed (e.g. because it is no longer in view, or due to a long occlusion), and afterwards the text entity it was tracking is detected again, it will be added again as a new tracker. We find this to be an adequate compromise for efficient and long periods of continuous operation.

4.1.4 Occlusions: These mechanisms allow our system to deal with brief occlusions of the tracked text regions. A full occlusion will produce bad tracks for the affected trackers. If the occlusion is shorter than the number of frames needed to delete a tracker, when the text region is in view again it will be recovered. The system will also be able to recover the track even with big translations or wide baseline changes of orientation thanks to the use of high level features. Partial occlusions of text regions will be also dealt with in the same fashion, and even on tracker creation, because of the feature addition mechanism.

4.2 OCR and speech synthesis

When a tracker is considered stable, it is then a candidate for recognition. The image quadrilateral enclosing the tracked text entity is rectified to a fronto-parallel view using the state homography transformation and then sent to OCR, for

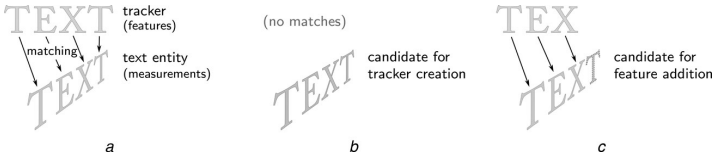


Fig. 6 *Tracker maintenance*

a Perfect tracking
b Tracker creation
c Feature addition

doi: 10.1049/iet-cvi.2013.0217

which we use the Tesseract OCR engine [http://code.google.com/p/tesseract-ocr/]. Recognition is performed in a parallel processing task, so the tracking is maintained in real-time while the recognition runs alongside (for implementation details see Section 4.3). The decision on which tracker to recognise is weighted by several factors: whether or not there is already any recognition available for this tracker, the OCR confidence value of previous recognitions, and the elapsed time since the last recognition attempt. A tracker might be sent to OCR for recognition several times, but if the returned confidence value of a new recognition is lower than a previous one, the recognition result with higher confidence is kept.

Speech synthesis is the main user interface of the system, and the main intended communication with the user. Those text regions that have a high enough OCR confidence value and have a stable text tracker are considered for being synthesised into speech. The text is sent to an speech synthesis engine (in our case we use Microsoft's speech API) so the recognised text is played back to the user. The candidate texts are queued and prioritised according to the distance to the centre of the image. Regions that stay in view of the camera for long enough might be reproduced several times, but as the region identity is maintained throughout the sequence, this delay can be adjusted for the convenience of the user, that is, by tracking the text we can avoid the system continuously repeating the same text over and over.

4.3 Implementation

One of the design objectives of our prototype is real-time operation. The system is carefully parallelised to make the best possible use of available processors. We use Intel's Threading Building Blocks (TBB) [http://threadingbuildingblocks.org/], which implements a task-based parallelisation paradigm. It features a high level C++ API for defining parallel constructions, supports nested parallelism and provides automatic scalability. The algorithm steps in Fig. 1 are implemented as separate stages of a processing pipeline. On multiprocessor machines, TBB is able to schedule different stages on different processors so several frames might be simultaneously processed at any given moment. The system maintains a global state and a transient state. The transient state is carried forward across the stages of the pipeline. At the end of a frame processing, the transient changes are atomically combined in the global system state. This is easily implemented as the library guarantees strict sequential ordering of the pipeline stages of consecutive frames (i.e. the tracking stage on frame n is guaranteed to run before the tracking stage of frame $n+1$). OCR is spawned as a task outside the main processing pipeline and thus it is scheduled concurrently with it. This allows to maintain real-time performance for the text tracking while being able to perform longer running processes at the same time and without under- or over-subscribing the available parallelism. We find this to be a superior design in terms of portability and scalability when compared with other approaches (such as e.g. PTAM [33]) in which explicit threads are defined with synchronisation mechanisms between them.

5 Results and discussion

To demonstrate and validate the proposed system, at first we present a quantitative analysis of the text tracking mechanism based on standardised metrics and annotated ground-truth data that help establish a performance baseline for comparative studies. Then, a qualitative evaluation of the prototype's operation in everyday scenarios is outlined to provide an insight into the future improvements and requirements of a text reading system. For our quantitative experiments, three challenging video sequences are used: HOSPITAL, MERCHANT and QUEEN. These contain scene text in a city environment and suffer from hand-held erratic camera motion, as well as blur and a great degree of perspective distortion. The sequences were annotated to obtain a (i) ground-truth labelling of text, (ii) 3D bounding quadrilaterals and (iii) region identity between frames [The video sequences and the ground-truth labelling are available at http://nf.ull.es/q/texttrack]. To achieve these, tracking was applied in the video sequences using a commercial match-moving software, with each video requiring extensive manual adjustment of the tracked features. After that, 3D editing software was used to locate rectangles in the 3D space around the projected positions of the text in the scene. When the rectangles are projected back as quadrilaterals into the image plane, they perfectly fit the text as seen in the image. The total number of annotated frames is 930. For the qualitative analysis, a variety of example videos, showcasing different scenarios, were experimented with as shown later.

5.1 Quantitative analysis of the tracking mechanism

Two distinct tests were performed to evaluate the two desired characteristics of a text tracking system: (i) increase in segmentation accuracy and (ii) the ability to maintain region identity. The first test is a frame-by-frame comparison of the text detection accuracy between the text segmentation stage alone against the segmented text regions while tracking by our method. The second test evaluates the tracking performance by measuring the detection accuracy along with the region identity across the sequence as a whole.

5.1.1 Frame-by-frame evaluation: For our text segmentation evaluation, we use the precision and recall measures as introduced in the ICDAR 2003 Robust Reading Competition [5], slightly adapted to operate on arbitrary quadrilaterals instead of rectangles. The degree of match between two quadrilaterals $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ is defined as

$$\text{match}(\boldsymbol{q}_1, \boldsymbol{q}_2) = \frac{\text{area}(\boldsymbol{q}_1 \cap \boldsymbol{q}_2)}{\text{area}(\boldsymbol{q}_1 \cup \boldsymbol{q}_2)} \tag{12}$$

When comparing a quadrilateral $\boldsymbol{q}$ against a set of quadrilaterals Q, the best match is computed as

$$\text{bestmatch}(\boldsymbol{q}, Q) = \max_{\boldsymbol{q}' \in Q} \text{match}(\boldsymbol{q}, \boldsymbol{q}') \tag{13}$$

Then, the precision p, recall r and f measures for a certain frame are defined as

$$p = \frac{\sum_{\boldsymbol{q} \in E} \text{bestmatch}(\boldsymbol{q}, G)}{|E|} \tag{14}$$

$$r = \frac{\sum_{\boldsymbol{q} \in G} \text{bestmatch}(\boldsymbol{q}, E)}{|G|} \tag{15}$$

$$f = \frac{2}{1/p + 1/r} \tag{16}$$

where G is the set of quadrilaterals in the ground-truth and E is the set of quadrilaterals being evaluated.

These measures were computed for each one of the frames in the sequences, comparing the quadrilaterals produced by the text segmentation and orientation detection stages with the quadrilaterals produced by the tracking stage. In our evaluations, we are only considering text lines with four or more characters from the ground-truth, as this is the minimum length at which the perspective estimator works reliably [7]. The results are shown in Figs. 7*b*, 8*b* and 9*b*, where, for every frame in the sequences, the f measure for the segmentation and tracking outputs are plotted.

In the HOSPITAL sequence (Fig. 7), the camera approaches a road sign with strong perspective distortion being introduced gradually, and featuring a very textured tree background, which makes text segmentation challenging, as can be seen in the performance of the text segmenter. The accuracy of the tracked regions is very good throughout the sequence even when the perspective distortion is significant. The continuous change in perspective occasionally causes the trackers to lose track as the filters converge into the new orientation. This can be seen in the dips around frames 5, 20 and especially between frames 100 and 115 and at the end of the sequence. Nevertheless, the region identity is never lost, and the trackers recover the lock on the text shortly afterwards.

In the MERCHANT sequence (Fig. 8), a street sign is panned laterally, with a wide baseline change of orientation and a textured regular brick pattern in the background. Text region tracking is accurate across most of the sequence, as shown in the results. Extreme camera shake is the cause for some of the individual trackers to momentarily lose track, from frame 125 to the end. The system does not produce a box for a text region that is not tracked with enough confidence, thus the

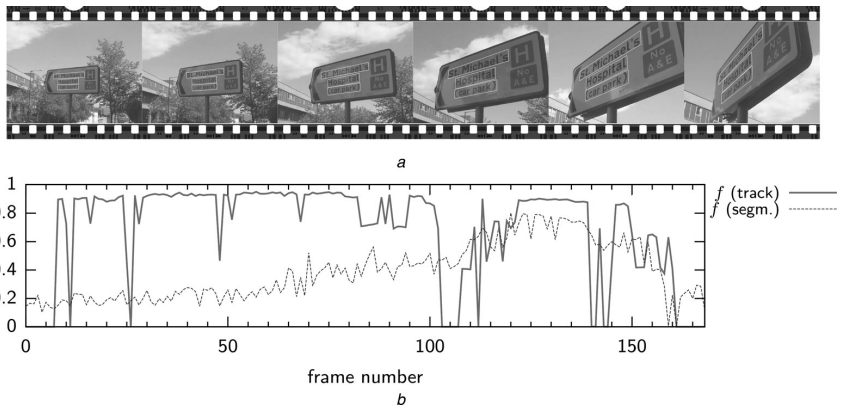


Fig. 7 *Video sequence HOSPITAL*

a Video sequence with tracked text bounding quadrilaterals
b Frame-by-frame segmentation accuracy

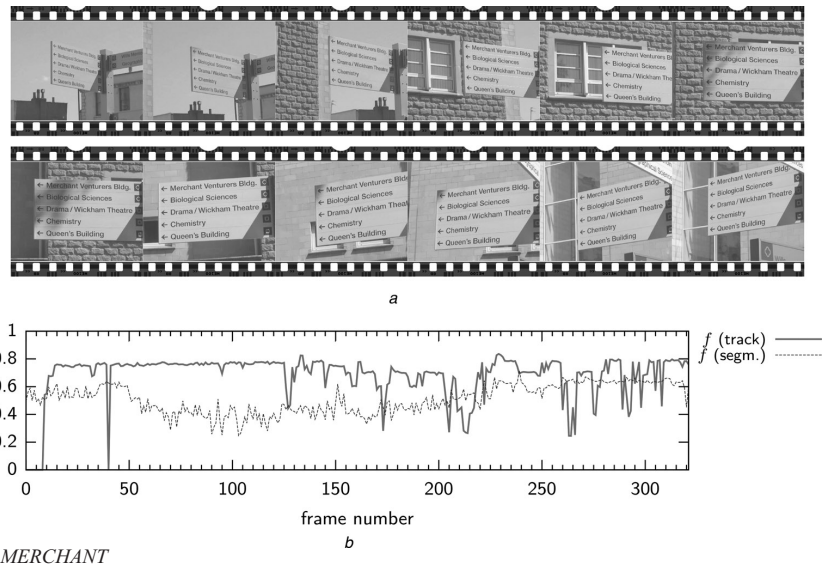


Fig. 8 *Video sequence MERCHANT*

a Video sequence with tracked text bounding quadrilaterals
b Frame-by-frame segmentation accuracy

IET Comput. Vis., pp. 1–12
doi: 10.1049/iet-cvi.2013.0217

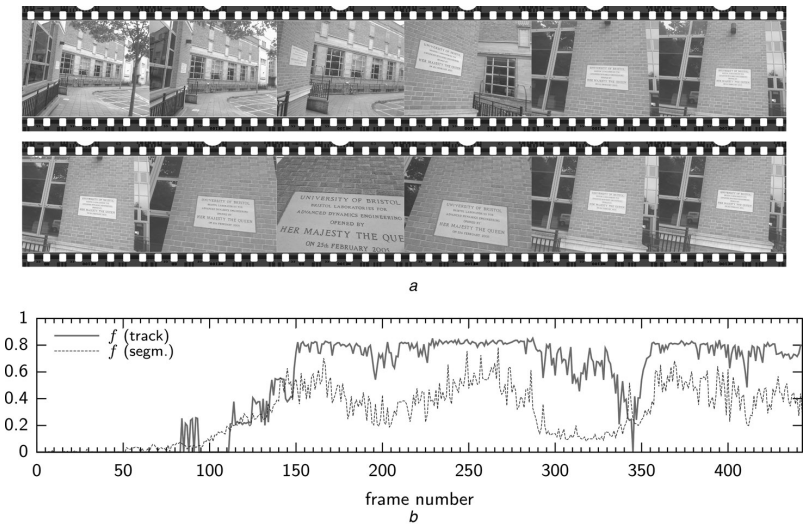


Fig. 9 *Video sequence QUEEN*

a Video sequence with tracked text bounding quadrilaterals
b Frame-by-frame segmentation accuracy

alternating and temporary disappearances of some of the boxes across the sequence. Those are promptly recovered without losing the region identity. This is also the reason for the dip in the graph around frame 40, where all the trackers are lost for just one frame.

Finally, the QUEEN sequence (Fig. 9) features a walkabout towards a building sign which is not visible from the start of the sequence, finishing with a close approach into the text. This very challenging sequence also shows regular window and railing patterns. As can be seen in the results, during the first part of the sequence – until frame 120 – the accuracy is 0 as there is no text. When the text is completely visible, the tracking quickly locks on the text lines and this is clearly shown in the graph between frames 120 and 280. Then, during a camera movement towards the text, there are two brief camera blur events between frames 280–300 and 330–350 that cause the segmentation to produce very few regions, and thus making the tracker to momentarily lose track as there are no regions to track, especially at frame 345. The tracker promptly recovers the track after these events. The ability to quickly recover from failures demonstrates the versatility of the proposed method. To illustrate the effect that text tracking has on segmentation accuracy, Fig. 10 shows the output of the text segmentation stage, with a considerable number of false positives, compared to the output of the tracking stage, where the spurious regions have been discarded.

5.1.2 Tracking evaluation: The characteristics of our prototype system preclude making a comparative study against any of the few other published text tracking systems. For example, in the work in [12], the text regions are manually selected prior to tracking or in the work in [34] text tracking was performed on overlaid text. Furthermore, we are tracking 3D text quadrilaterals, as opposed to 2D rectangles, which is a fundamental difference with any other published work (e.g. [16, 18]). With this paper, we are also publishing our scene text tracking dataset and its associated ground-truth data – something that has not been done before – in the hope that it will be useful to other researchers and to enable future comparative studies.

However, we do present comparative results against our own previous work in [8]. To evaluate the performance of the text tracking we adopt the CLEAR object tracking metrics as suggested by Bernardin and Stiefelhagen [35] and Kasturi *et al.* [34]. The metrics are designed to evaluate the detection and tracking performance across a sequence as a whole, and thus penalise false positives and false negatives as well as region identity changes or losses. For every frame k, there is a mapping M_k between the elements in the ground-truth and detected sets

$$M_k = \{(\boldsymbol{g}, \boldsymbol{e}),\ \text{with } \boldsymbol{g} \in G_k \text{ and } \boldsymbol{e} \in E_k\} \tag{17}$$

where G_k and E_k are the sets of ground-truth and detected quadrilaterals at frame k. Mappings are unique for each element on each set. Our criteria for considering a candidate match between two quadrilaterals is that they have some overlap (e.g. area$(\boldsymbol{q} \cap \boldsymbol{q}') > 0$). To select a unique match between the candidates, we first consider the same match as in the previous frame if they still overlap; otherwise the pair with maximum overlap is chosen (refer to [35] for the full

Fig. 10 *Comparison of the output of the*

a Text segmentation
b Tracking stages, for one frame in the QUEEN sequence

Table 1 Whole sequence tracking evaluation

Sequence	Proposed method		PF method [8]	
	MOTP	MOTA	MOTP	MOTA
HOSPITAL	0.89	0.7	0.1	0.32
MERCHANT	0.78	0.82	0.16	0.24
QUEEN	0.8	0.65	0.1	0.13

explanation and rationale of the matching strategy). Every detected quadrilateral that is not mapped is a false positive; likewise, every unmapped ground-truth quadrilateral is a missed detection. If a quadrilateral $\boldsymbol{g} \in G_k$ is matched to different quadrilaterals $\boldsymbol{q}, \boldsymbol{q}' \in E_k$ in consecutive frames, it is considered an identity mismatch.

Two measures are defined, the multiple object tracking precision (MOTP) and the multiple object tracking accuracy (MOTA)

$$\text{MOTP} = \frac{\sum_k \sum_{(\boldsymbol{g},\boldsymbol{e}) \in M_k} \text{match}(\boldsymbol{g}, \boldsymbol{e})}{\sum_k |M_k|} \tag{18}$$

$$\text{MOTA} = 1 - \frac{\sum_k \left(\delta_k + \phi_k + \log_{10}(\rho_k)\right)}{\sum_k |G_k|} \tag{19}$$

where δ_k, ϕ_k and ρ_k are the total number of missed detections, false positives and id mismatches for frame k, respectively.

Table 1 summarises the results of our proposed method in comparison to our previous text tracking technique using PFs [8]. Our PF method had a simple 2D state model and did not

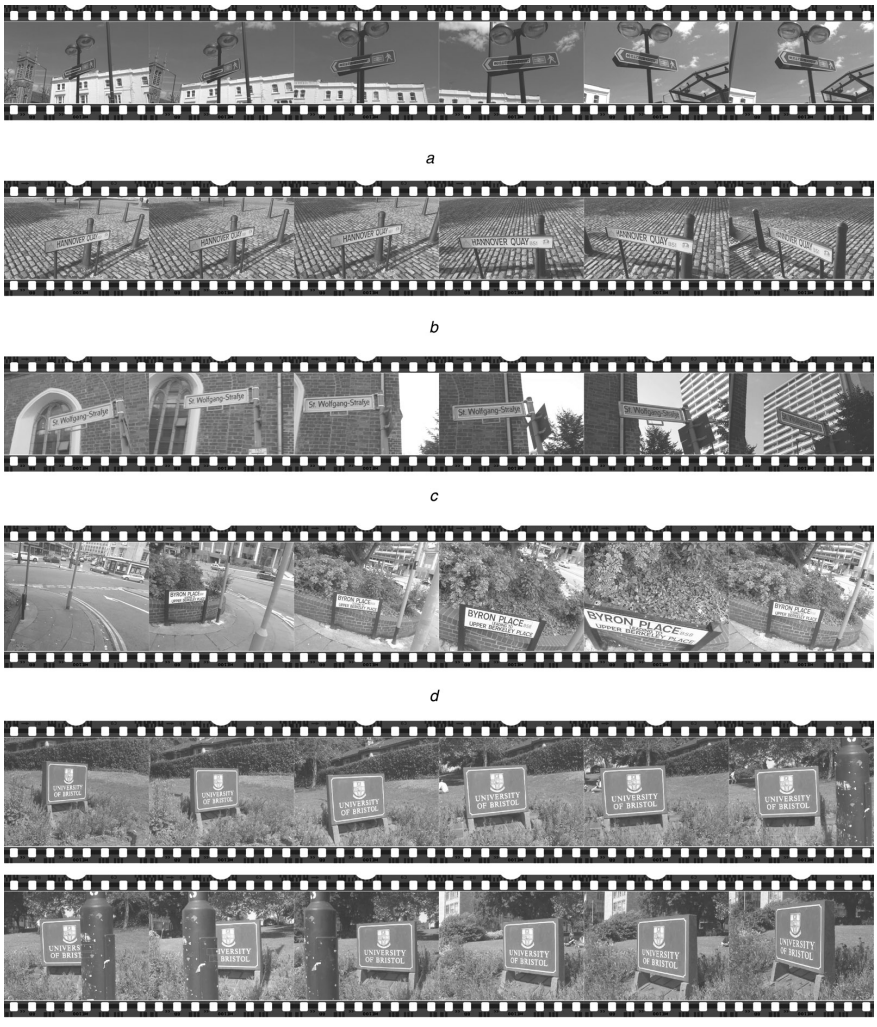


Fig. 11 *Video sequences from the qualitative evaluation experiment*

a Sequence CLIFTON
b Sequence HANNOVER
c Sequence WOLFGANG
d Sequence BYRON PLACE
e Sequence UOB

IET Comput. Vis., pp. 1–12
doi: 10.1049/iet-cvi.2013.0217

perform any perspective estimation and this is shown in the MOTP values, where the proposed method consistently achieves very high values (0.89, 0.80 and 0.78 for the HOSPITAL, QUEEN and MERCHANT sequences, respectively) thanks to the enhanced state model. It also produces high MOTA values (0.82, 0.70 and 0.59 for the MERCHANT, HOSPITAL and QUEEN sequences, respectively) because of the low number of false positives and id mismatches that the tracking produces. As previously explained, the challenging sequences used in our experiments feature very erratic camera motion, blur and perspective distortion. Our MOTA evaluation is penalised on those frames where the text is not tracked with enough confidence, in which our system does not produce any box, being counted as a missed detection.

5.2 Qualitative evaluation

We show more examples in Fig. 11 to further illustrate the operation of the proposed method. In the CLIFTON sequence (Fig. 11*a*), the text is never in a fronto-parallel, horizontal orientation with respect to the camera and undergoes a wide baseline change in orientation. This is also true for the WOLFGANG sequence (Fig. 11*c*), which also features a complex background and very blurred frames because of camera vibrations. The HANNOVER sequence (Fig. 11*b*) contains regular, very contrasted tiles in the background which produce a great number of false positives from the text segmentation stage. The BYRON PLACE sequence (Fig. 11*d*) features a change in orientation around elevation. Note, the text is not visible at the start of the sequence, but as soon as it appears, the system is able to pick up the location of the various lines of text and then track them continuously. As with previous sequences (i.e. MERCHANT and QUEEN), camera shake is responsible for the momentary disappearance of tracked regions, after which the tracker recovers without region identity loss. Finally, the UOB sequence (Fig. 11*e*) features a moving partial occlusion across the tracked text. As there are always enough visible features (i.e. characters) for each one of the text lines, the proposed method is able to keep the identity and location of all the text regions in the scene. For all the sequences, the system is able to quickly spot the text in its original orientation and maintain the region identity throughout the duration of the video.

5.3 Performance

The system operates at video rate (average 25 fps) on the video sequences used for our experiments. They were measured on a standard PC with an Intel Core 2 Quad Q6600 processor and 8 Gb of RAM. A breakdown of the times spent by the algorithm on each of the stages is presented in Table 2. The most expensive stage in terms of computation time is the tracking observation, which includes the feature matching. The text segmentation and perspective estimation are also major contributors to the time needed to process one frame. Those stages are split into a pipeline and are automatically distributed between the processor cores to achieve parallelism. The OCR task is comparatively much slower than the rest of the stages together. This demonstrates one of the benefits of text tracking: OCR runs on an independent thread and thus does not contest with the main processing pipeline to achieve the desired frame rate. When the recognition results are ready, the region identity maintained by the text tracking is used to assign the recognised text to the correct text region.

Table 2 Time spent on each stages of the algorithm

acquisition	5 ms
segmentation	22 ms
aggregation	5 ms
perspective estimation	27 ms
tracking: prediction	8 ms
tracking: observation	42 ms
OCR	250 ms

6 Conclusions

We have presented a text reading system based on text tracking. It operates autonomously and in real-time, automatically detecting and recognising new text regions and discarding the old ones. We have shown quantitative and qualitative analysis of the performance and capabilities and of our prototype. We are also releasing our scene text tracking dataset and its associated ground-truth data to enable future comparative studies.

The area of text tracking is very young and there is still a lot to be accomplished. Although we feel we have presented a novel step towards fast text segmentation and tracking, there remain a number of shortcomings and newer goals that are yet to be addressed. Our method is focused on larger text and is not suited to deal with smaller document texts. Also, as a matter of trading accuracy for speed, we do not necessarily use state-of-the-art text segmentation. There remain other avenues to explore, for example, (a) exploiting the OCR results from multiple frames to combine and obtain a more accurate global recognition (e.g. to bypass reflections and occlusions), (b) combining rectified images from several frames to help construct super-resolution and/or larger (by mosaicking) images. Moreover, we still have to study and understand how the tracking information can help build a better user interface for assistive devices. Observing the patterns of movement and context in the surroundings is crucial for deciding when and how to read text back to the user, enabling a more useful interaction experience. We plan to develop these ideas as part of our future work.

7 Acknowledgments

This work was carried out at Bristol University by Carlos Merino-Gracia, who is funded by the Spanish Ministerio de Industria y Comercio (project IPT-2012-0961-300000).

8 References

1 Epshtein, B., Ofek, E., Wexler, Y.: 'Detecting text in natural scenes with stroke width transform'. Computer Vision and Pattern Recognition, 2010, pp. 2963–2970
2 Pan, Y.-F., Hou, X., Liu, C.-L.: 'A hybrid approach to detect and localize texts in natural scene images', *Trans. Image Process.*, 2011, **20**, (3), pp. 800–813
3 Neumann, L., Matas, J.: 'Real-time scene text localization and recognition'. Computer Vision and Pattern Recognition, 2012, pp. 3538–3545
4 Chen, H., Tsai, S., Schroth, G., Chen, D., Grzeszczuk, R., Girod, B.: 'Robust text detection in natural images with edge-enhanced maximally stable external regions'. Int. Conf. on Image Processing, 2011, pp. 2609–2612

5 Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: 'ICDAR 2003 robust reading competitions'. Int. Conf. on Document Analysis and Recognition, 2003, pp. 682–687
6 Myers, G.K., Bolles, R.C., Luong, Q.-T., Herson, J.A., Aradhye, H.B.: 'Rectification and recognition of text in 3-D scenes', *Int. J. Doc. Anal. Recognit.*, 2005, **7**, pp. 147–158
7 Merino-Gracia, C., Mirmehdi, M., Sigut, J., González-Mora, J.L.: 'Fast perspective recovery of text in natural scenes', *Image Vis. Comput.*, 2013, **31**, pp. 714–724
8 Merino, C., Mirmehdi, M.: 'A framework towards realtime detection and tracking of text'. Camera Based Document Analysis and Recognition, 2007, pp. 10–17
9 Li, H., Doermann, D., Kia, O.: 'Automatic text detection and tracking in digital video', *Trans. Image Process.*, 2000, **9**, (1), pp. 147–156
10 Lienhart, R., Wernicke, A.: 'Localizing and segmenting text in images and videos', *Circuits Syst. Video Technol.*, 2002, **12**, (4), pp. 256–268
11 Gllavata, J., Ewerth, R., Freisleben, B.: 'Tracking text in MPEG videos'. Int. Conf. on Multimedia, 2004, pp. 240–243
12 Myers, G.K., Burns, B.: 'A robust method for tracking scene text in video imagery'. Camera Based Document Analysis and Recognition, 2005, vol. 1
13 Shiratori, H., Goto, H., Kobayashi, H.: 'An efficient text capture method for moving robots using DCT feature and text tracking'. Int. Conf. on Pattern Recognition, 2006, pp. 1050–1053
14 Tanaka, M., Goto, H.: 'Autonomous text capturing robot using improved DCT feature and text tracking'. Int. Conf. on Document Analysis and Recognition, 2007, 2, pp. 1178–1182
15 Tanaka, M., Goto, H.: 'Text-tracking wearable camera system for visually-impaired people'. Int. Conf. on Pattern Recognition, 2008, pp. 1–4
16 Goto, H., Tanaka, M.: 'Text-tracking wearable camera system for the blind'. Int. Conf. on Document Analysis and Recognition, 2009, pp. 141–145
17 Na, Y., Wen, D.: 'An effective video text tracking algorithm based on sift feature and geometric constraint'. Advances in Multimedia Information Processing, 2010, pp. 392–403
18 Minetto, R., Thome, N., Cord, M., Leite, N.J., Stolfi, J.: 'Snoopertrack: text detection and tracking for outdoor videos'. Int. Conf. on Image Processing, 2011, pp. 505–508
19 Lowe, D.G.: 'Distinctive image features from scale-invariant keypoints', *Int. J. Comput. Vis.*, 2004, **60**, (2), pp. 91–110
20 Hartley, R.I., Zisserman, A.: 'Multiple view geometry in computer vision' (Cambridge University Press, 2004, 2nd edn.)
21 Phan, T.Q., Shivakumara, P., Lu, T., Tan, C.L.: 'Recognition of video text through temporal integration'. Int. Conf. on Document Analysis and Recognition, 2013, pp. 589–593
22 Mosleh, A., Bouguila, N., Hamza, A.: 'Automatic inpainting scheme for video text detection and removal', *Trans. Image Process.*, 2013, **22**, (11), pp. 4460–4472
23 Wang, K., Babenko, B., Belongie, S.: 'End-to-end scene text recognition'. Int. Conf. on Computer Vision, 2011, pp. 1457–1464
24 Mishra, A., Alahari, K., Jawahar, C.: 'Top-down and bottom-up cues for scene text recognition'. Computer Vision and Pattern Recognition, 2012, pp. 2687–2694
25 González, A., Bergasa, L.M.: 'A text reading algorithm for natural images', *Image Vis. Comput.*, 2013, **31**, (3), pp. 255–274
26 Merino-Gracia, C., Lenc, K., Mirmehdi, M.: 'A head-mounted device for recognizing text in natural scenes', in Iwamura, M., Shafait, F. (Eds.): 'Camera based document analysis and recognition', 2012, (*LNCS*, **7139**), pp. 29–41
27 Targhi, A., Hayman, E., Eklundh, J., Shahshahani, M.: 'The Eigen-transform & applications'. Asian Conf. of Computer Vision, I, 2006, pp. 70–79
28 de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: 'Computational geometry: algorithms and applications' (Springer-Verlag, 2000, 2nd edn.)
29 Pilu, M.: 'Extraction of illusory linear clues in perspectively skewed documents'. Computer Vision and Pattern Recognition, 2001, pp. 363–368
30 Toussaint, G.: 'Solving geometric problems with the rotating calipers'. Mediterranean Electrotechnical Conf., 1983, pp. 10–17
31 Doucet, A., de Freitas, J., Gordon, N.: 'Sequential Monte Carlo methods in practice' (Springer-Verlag, 2001)
32 Wan, E., Van Der Merwe, R.: 'The unscented Kalman filter for nonlinear estimation'. Adaptive Systems for Signal Processing, Communications, and Control, 2000, pp. 153–158
33 Klein, G., Murray, D.: 'Parallel tracking and mapping for small AR workspaces'. ISMAR, 2007, pp. 225–234
34 Kasturi, R., Goldgof, D., Soundararajan, P., *et al.*: 'Framework for performance evaluation of face, text, and vehicle detection and tracking in video: data, metrics, and protocol', *Pattern Anal. Mach. Intell.*, 2009, **31**, (2), pp. 319–336
35 Bernardin, K., Stiefelhagen, R.: 'Evaluating multiple object tracking performance: the CLEAR MOT metrics', *J. Image Video Process.*, 2008, **2008**, pp. 1:1–1:10

9 Appendix: Homography decomposition

As noted by Hartley and Zisserman [20], a homography can be decomposed into three partial transformations $\boldsymbol{H}_{\mathrm{S}}$, $\boldsymbol{H}_{\mathrm{A}}$ and $\boldsymbol{H}_{\mathrm{P}}$, each of them representing a separate 'similarity', 'affinity' and 'projectivity', respectively

$$\boldsymbol{H} = \boldsymbol{H}_{\mathrm{S}}\boldsymbol{H}_{\mathrm{A}}\boldsymbol{H}_{\mathrm{P}} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0^{\top} & 1 \end{bmatrix}\begin{bmatrix} \boldsymbol{SK} & 0 \\ 0^{\top} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & 0 \\ \boldsymbol{l}^{\top} & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{t} \\ \boldsymbol{l}^{\top} & 1 \end{bmatrix} \tag{20}$$

where $\boldsymbol{A}$ is a non-singular matrix defined as $\boldsymbol{A} = \boldsymbol{RSK} + \boldsymbol{tl}^{\mathrm{T}}$, $\boldsymbol{R}$ is a rotation matrix, $\boldsymbol{S}$ is an anisotropic scaling matrix, $\boldsymbol{K}$ is a shear matrix, $\boldsymbol{hbft}$ is a translation vector and $\boldsymbol{l}$ is a perpsective foreshortening vector. They are further defined as

$$\boldsymbol{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$
$$\boldsymbol{K} = \begin{bmatrix} 1 & \sigma \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \text{ and } \boldsymbol{l} = \begin{bmatrix} l_x \\ l_y \end{bmatrix} \tag{21}$$

With (20) and (21) the function $\boldsymbol{H} = \mathcal{H}(\boldsymbol{h})$ is completely specified. When given the homography $\boldsymbol{H}$, defined as

$$\boldsymbol{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

the four direct parameters are trivially obtained as

$$t_x = h_{13}, \quad t_y = h_{23}, \quad l_x = h_{31}, \quad l_y = h_{32} \tag{22}$$

then we define four auxiliary variables

$$g_1 = h_{11} - t_x v_x, \quad g_3 = h_{12} - t_x v_y \tag{23}$$

$$g_2 = h_{21} - t_y v_x, \quad g_4 = h_{22} - t_y v_y \tag{24}$$

to be able to obtain the four remaining parameters as

$$\theta = \arctan\left(\frac{g_2}{g_1}\right), \quad \sigma = \frac{g_1 g_3 + g_2 g_4}{g_1^2 + g_2^2} \tag{25}$$

$$s_x = +\sqrt{g_1^2 + g_2^2}, \quad s_y = \frac{g_1 g_4 - g_2 g_3}{+\sqrt{g_1^2 + g_2^2}} \tag{26}$$

that completely specify the inverse function $\boldsymbol{h} = \mathcal{H}^{-1}(\boldsymbol{H})$.

doi: 10.1049/iet-cvi.2013.0217

References

[1] M. de Berg, M. van Kreveld, M. Overmars and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Second Edition. Springer-Verlag, 2000, p. 367 (cit. on p. 18).

[2] K. Bernardin and R. Stiefelhagen. 'Evaluating multiple object tracking performance: the CLEAR MOT metrics'. In: *Journal of Image Video Processing* 1 (Jan. 2008), pp. 1–10 (cit. on p. 47).

[3] E. Bertucci, M. Pilu and M. Mirmehdi. 'Text selection by structured light marking for hand-held cameras'. In: *International Conference on Document Analysis and Recognition*. Vol. 1. Aug. 2003, pp. 555–559 (cit. on p. 3).

[4] K. Bouman, G. Abdollahian, M. Boutin and E. Delp. 'A Low Complexity Sign Detection and Text Localization Method for Mobile Applications'. In: *IEEE Transactions on Multimedia* 13.5 (Oct. 2011), pp. 922–934 (cit. on p. 6).

[5] J. Canny. 'A Computational Approach to Edge Detection'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8.6 (Nov. 1986), pp. 679–698 (cit. on p. 5).

[6] D. Chen and J. Luettin. *A Survey of Text Detection and Recognition in Images and Videos*. Research Report 38-2000. IDIAP, 2000 (cit. on p. 4).

[7] H. Chen, S. Tsai, G. Schroth, D. Chen, R. Grzeszczuk and B. Girod. 'Robust text detection in natural images with edge-enhanced Maximally Stable Extremal Regions'. In: *International Conference on Image Processing*. Sept. 2011, pp. 2609–2612 (cit. on p. 5).

[8] J. Chmiel, O. Stankiewicz, W. Switala, M. Tluczek and J. Jelonek. *Read IT Project Report: A portable text reading system for the blind people*. Tech. rep. 2005 (cit. on p. 5).

[9] P. Clark and M. Mirmehdi. 'Estimating the Orientation and Recovery of Text Planes in a Single Image'. In: *British Machine Vision Conference*. Sept. 2001, pp. 421–430 (cit. on p. 10).

[10] P. Clark and M. Mirmehdi. 'Recognising text in real scenes'. In: *International Journal on Document Analysis and Recognition* 4 (2002), pp. 243–257 (cit. on p. 3).

[11] P. Clark and M. Mirmehdi. 'Rectifying perspective views of text in 3D scenes using vanishing points.' In: *Pattern Recognition* 36.11 (2003), pp. 2673–2686 (cit. on p. 10).

[12] W. Dong, Z. Lian, Y. Tang and J. Xiao. 'Text Detection in Natural Images Using Localized Stroke Width Transform'. In: *MultiMedia Modeling*. Ed. by X. He, S. Luo, D. Tao, C. Xu, J. Yang and M. Hasan. Vol. 8935. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 49–58 (cit. on p. 5).

[13] M. Donoser and H. Bischof. 'Efficient Maximally Stable Extremal Region (MSER) Tracking'. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2006, pp. 553–560 (cit. on p. 16).

[14] M. Donoser, C. Arth and H. Bischof. 'Detecting, Tracking and Recognizing License Plates'. In: *Asian Conference of Computer Vision*. 2007, pp. 447–456 (cit. on p. 5).

[15] A. Doucet, J. de Freitas and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001 (cit. on p. 25).

[16] J. Du, Q. Huo, L. Sun and J. Sun. 'Snap and Translate Using Windows Phone'. In: *International Conference on Document Analysis and Recognition*. Sept. 2011, pp. 809–813 (cit. on p. 6).

[17] B. Epshtein, E. Ofek and Y. Wexler. 'Detecting text in natural scenes with stroke width transform'. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2010, pp. 2963–2970 (cit. on pp. 4–5).

[18] N. Ezaki, M. Bulacu and L. Schomaker. 'Text Detection from Natural Scene Images: Towards a System for Visually Impaired Persons'. In: *International Conference on Pattern Recognition*. 2004, pp. 683–686 (cit. on p. 5).

[19] M. A. Fischler and R. C. Bolles. 'Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography'. In: *Communications of the ACM* 24.6 (June 1981), pp. 381–395 (cit. on pp. xiii, 21).

[20] E. E. Fournier d'Albe. 'On a Type-Reading Optophone'. In: *Royal Society of London Proceedings Series A* 90 (July 1914), pp. 373–375 (cit. on p. 2).

[21] L. Gómez and D. Karatzas. 'Scene Text Recognition: No Country for Old Men?' In: *Asian Conference of Computer Vision*. Ed. by C. Jawahar and S. Shan. Vol. 9009. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 157–168 (cit. on p. 5).

[22] H. Goto and M. Tanaka. 'Text-Tracking Wearable Camera System for the Blind'. In: *International Conference on Document Analysis and Recognition*. July 2009, pp. 141–145 (cit. on pp. 6, 47).

[23] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Second Edition. Cambridge University Press, 2004 (cit. on pp. 6, 9, 21, 26).

[24] T. Hedgpeth, J. A. Black and S. Panchanathan. 'A demonstration of the iCARE portable reader'. In: *ACM's Special Interest Group on Accessible Computing*. 2006, pp. 279–280 (cit. on p. 5).

[25] W. Huang, Z. Lin, J. Yang and J. Wang. 'Text Localization in Natural Images Using Stroke Feature Transform and Text Covariance Descriptors'. In: *IEEE International Conference on Computer Vision*. Dec. 2013, pp. 1241–1248 (cit. on p. 5).

[26] W. Huang, Y. Qiao and X. Tang. 'Robust Scene Text Detection with Convolution Neural Network Induced MSER Trees'. In: *European Conference on Computer Vision*. Ed. by D. Fleet, T. Pajdla, B. Schiele and T. Tuytelaars. Vol. 8692. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 497–511 (cit. on p. 5).

[27] K. Jung, K. I. Kim and A. K. Jain. 'Text information extraction in images and video: a survey'. In: *Pattern Recognition* 37.5 (2004), pp. 977–997 (cit. on p. 4).

[28] D. Karatzas, S. Robles Mestre, J. Mas, F. Nourbakhsh and P. Pratim Roy. 'ICDAR 2011 Robust Reading Competition - Challenge 1: Reading Text in Born-Digital Images (Web and Email)'. In: *International Conference on Document Analysis and Recognition*. Sept. 2011, pp. 1485–1490 (cit. on p. 33).

[29] D. Karatzas et al. 'ICDAR 2013 Robust Reading Competition'. In: *International Conference on Document Analysis and Recognition*. Aug. 2013, pp. 1484–1493 (cit. on p. 4).

[30] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova and J. Zhang. 'Framework for Performance Evaluation of Face, Text, and Vehicle Detection and Tracking in Video: Data, Metrics, and Protocol'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.2 (2009), pp. 319–336 (cit. on p. 47).

[31] G. Klein and D. Murray. 'Parallel Tracking and Mapping for Small AR Workspaces'. In: *International Symposium on Mixed and Augmented Reality*. Nov. 2007, pp. 225–234 (cit. on p. 31).

[32] G. Kramer. *Auditory Display: Sonification, Audification, and Auditory Interfaces*. Perseus Publishing, 1993 (cit. on p. 2).

[33] R. Kurzweil. *The age of spiritual machines: when computers exceed human intelligence*. Viking Press, 1998 (cit. on p. 3).

[34] J. Liang, D. Doermann and H. Li. 'Camera-based analysis of text and documents: a survey'. In: *International Journal on Document Analysis and Recognition* (2005), pp. 84–104 (cit. on p. 4).

[35] X. Lin. 'Reliable OCR solution for digital content re-mastering'. In: *Society of Photo-Optical Instrumentation Engineers*. Vol. 4670. 2001, pp. 223–231 (cit. on p. 3).

[36] D. G. Lowe. 'Distinctive Image Features from Scale-Invariant Keypoints'. In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110 (cit. on pp. xiii, 6, 8).

[37] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong and R. Young. 'ICDAR 2003 Robust Reading Competitions'. In: *International Conference on Document Analysis and Recognition*. 2003, pp. 682–687 (cit. on pp. 4, 8, 33, 41).

[38] S. Lucas. 'ICDAR 2005 text locating competition results'. In: *International Conference on Document Analysis and Recognition*. 2005, pp. 80–84 (cit. on p. 4).

[39] C. Mancas-Thillou, S. Ferreira, J. Demeyer, C. Minetti and B. Gosselin. 'A multifunctional reading assistant for the visually impaired'. In: *Journal of Image Video Processing* 2007.3 (Nov. 2007), pp. 1–11 (cit. on p. 5).

[40] C. Mancas-Thillou and M. Mirmehdi. 'An Introduction to Super-Resolution Text'. In: *Digital Document Processing*. Ed. by B. Chaudhuri. Advances in Pattern Recognition. Springer London, 2007, pp. 305–327 (cit. on p. 7).

[41] J. Matas, O. Chum, M. Urban and T. Pajdla. 'Robust wide baseline stereo from maximally stable extremal regions.' In: *British Machine Vision Conference*. 2002, pp. 384–396 (cit. on pp. xiii, 5, 16).

[42] W. W. Mayol, B. J. Tordoff and D. W. Murray. 'Wearable Visual Robots'. In: *Personal and Ubiquitous Computing* 6 (2002), pp. 37–48 (cit. on p. 31).

[43] C. Merino and M. Mirmehdi. 'A Framework Towards Realtime Detection and Tracking of Text'. In: *Camera Based Document Analysis and Recognition*. 2007, pp. 10–17 (cit. on pp. 7, 13, 18, 25, 33, 47–48). The full text is available on Appendix A.

[44] C. Merino-Gracia and M. Mirmehdi. 'Real-time text tracking in natural scenes'. In: *IET Computer Vision* 8.6 (2014), pp. 670–681 (cit. on pp. 7, 10, 13, 25–26, 33). The full text is available on Appendix E.

[45] C. Merino-Gracia, K. Lenc and M. Mirmehdi. 'A Head-Mounted Device for Recognizing Text in Natural Scenes'. In: *Camera Based Document Analysis and Recognition*. Ed. by M. Iwamura and F. Shafait. Vol. 7139. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2012, pp. 29–41 (cit. on pp. 5, 7–8, 13, 33). The full text is available on Appendix C.

[46] C. Merino-Gracia, M. Mirmehdi, J. Sigut and J. L. González-Mora. 'Fast perspective recovery of text in natural scenes'. In: *Image and Vision Computing* 31.10 (2013), pp. 714–724 (cit. on pp. 7, 10, 13, 18, 20, 33, 44). The full text is available on Appendix D.

[47] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. V. Gool. 'A Comparison of Affine Region Detectors'. In: *International Journal of Computer Vision* 65.1-2 (Nov. 2005), pp. 43–72 (cit. on p. 8).

[48] R. Minetto, N. Thome, M. Cord, N. J. Leite and J. Stolfi. 'Snoopertrack: Text detection and tracking for outdoor videos'. In: *International Conference on Image Processing*. Sept. 2011, pp. 505–508 (cit. on pp. 6, 47).

[49] M. Mirmehdi, P. Clark and J. Lam. 'A non-contact method of capturing low-resolution text for OCR'. In: *Pattern Analysis & Applications* 6.1 (2003), pp. 12–21 (cit. on p. 3).

[50] M. Mirmehdi, X. Xie and J. Suri. *Handbook of Texture Analysis*. Imperial College Press, 2008 (cit. on p. 15).

[51] A. Mishra, K. Alahari and C. Jawahar. 'Top-down and bottom-up cues for scene text recognition'. In: *IEEE Conference on Computer Vision and Pattern Recognition*. June 2012, pp. 2687–2694 (cit. on p. 5).

[52] S. Mori, C. Suen and K. Yamamoto. 'Historical review of OCR research and development'. In: *Proceedings of the IEEE* 80.7 (July 1992), pp. 1029–1058 (cit. on p. 3).

[53] A. Mosleh, N. Bouguila and A. B. Hamza. 'Image Text Detection Using a Bandlet-Based Edge Detector and Stroke Width Transform.' In: *British Machine Vision Conference*. 2012, pp. 1–12 (cit. on p. 5).

[54] 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'. In: *International Journal on Document Analysis and Recognition* 7.2 (2005), pp. 147–158 (cit. on pp. 9–10, 33–37, 40–41).

[55] G. K. Myers and B. Burns. 'A Robust Method for Tracking Scene Text in Video Imagery'. In: *Camera Based Document Analysis and Recognition*. 2005 (cit. on p. 47).

[56] Y. Na and D. Wen. 'An effective video text tracking algorithm based on SIFT feature and geometric constraint'. In: *Advances in Multimedia Information Processing*. 2010, pp. 392–403 (cit. on p. 6).

[57] G. Nagy. 'Twenty years of document image analysis in PAMI'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.1 (Jan. 2000), pp. 38–62 (cit. on p. 3).

[58] L. Najman and M. Couprie. 'Building the Component Tree in Quasi-Linear Time'. In: *IEEE Transactions on Image Processing* 15.11 (2006), pp. 3531–3539 (cit. on p. 16).

[59] L. Neumann and J. Matas. 'Real-time scene text localization and recognition'. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 3538–3545 (cit. on p. 5).

[60] L. Neumann and J. Matas. 'Text Localization in Real-World Images Using Efficiently Pruned Exhaustive Search'. In: *International Conference on Document Analysis and Recognition*. Sept. 2011, pp. 687–691 (cit. on p. 5).

[61] L. Neumann and J. Matas. 'A method for text localization and recognition in real-world images'. In: *Asian Conference of Computer Vision*. 2010, pp. 257–261 (cit. on p. 5).

[62] W. Newman, C. Dance, A. Taylor, S. Taylor, M. Taylor and T. Aldhous. 'CamWorks: a video-based tool for efficient capture from paper source documents'. In: *IEEE International Conference on Multimedia Computing and Systems*. Vol. 2. July 1999, pp. 647–653 (cit. on p. 3).

[63] D. Nistér and H. Stewénius. 'Linear Time Maximally Stable Extremal Regions'. In: *European Conference on Computer Vision*. 2008, pp. 183–196 (cit. on p. 17).

[64] M. Petter, V. Fragoso, M. Turk and C. Baur. 'Automatic text detection for mobile augmented reality translation'. In: *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. Nov. 2011, pp. 48–55 (cit. on p. 6).

[65] T. Q. Phan, P. Shivakumara, T. Lu and C. L. Tan. 'Recognition of Video Text through Temporal Integration'. In: *International Conference on Document Analysis and Recognition*. 2013, pp. 589–593 (cit. on p. 6).

[66] M. Pilu. 'Extraction of illusory linear clues in perspectively skewed documents'. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2001, pp. 363–368 (cit. on pp. 18–19).

[67] A. Rodríguez-Hernández, C. Merino, O. Casanova, C. Modroño, M. Torres, R. Montserrat, G. Navarrete, E. Burunat and J. González-Mora. 'Sensory substitution for visually disabled people: Computer solutions'. In: *WSEAS Transactions on Biology and Biomedicine* 7.1 (2010), pp. 1–10 (cit. on pp. 7–8, 13, 33). The full text is available on Appendix B.

[68] T. Saba, G. Sulong and A. Rehman. 'Document image analysis: issues, comparison of methods and remaining problems'. In: *Artificial Intelligence Review* 35.2 (2011), pp. 101–118 (cit. on p. 3).

[69] H. F. Schantz. *History of OCR, Optical Character Recognition*. Recognition Technologies Users Association, 1982 (cit. on p. 3).

[70] M. R. Schroeder. 'A brief history of synthetic speech'. In: *Speech Communication* 13.1–2 (1993), pp. 231–237 (cit. on p. 3).

[71] A. Shahab, F. Shafait and A. Dengel. 'ICDAR 2011 Robust Reading Competition Challenge 2: Reading Text in Scene Images'. In: *International Conference on Document Analysis and Recognition*. Sept. 2011, pp. 1491–1496 (cit. on pp. 4, 6, 33).

[72] H. Shiratori, H. Goto and H. Kobayashi. 'An Efficient Text Capture Method for Moving Robots Using DCT Feature and Text Tracking'. In: *International Conference on Pattern Recognition*. 2006, pp. 1050–1053 (cit. on p. 6).

[73] R. Smith. 'An Overview of the Tesseract OCR Engine'. In: *International Conference on Document Analysis and Recognition*. 2007, pp. 629–633 (cit. on pp. 31, 34).

[74] M. Tanaka and H. Goto. 'Autonomous Text Capturing Robot Using Improved DCT Feature and Text Tracking'. In: *International Conference on Document Analysis and Recognition*. Vol. 2. Sept. 2007, pp. 1178–1182 (cit. on p. 6).

[75] M. Tanaka and H. Goto. 'Text-tracking wearable camera system for visually-impaired people'. In: *International Conference on Pattern Recognition*. Dec. 2008, pp. 1–4 (cit. on p. 6).

[76] Y. Y. Tang, S.-W. Lee and C. Y. Suen. 'Automatic document processing: A survey'. In: *Pattern Recognition* 29.12 (1996), pp. 1931–1952 (cit. on p. 3).

[77] A. T. Targhi, E. Hayman and J.-o. Eklundh. 'Real-Time Texture Detection Using the LU-Transform'. In: *Computation Intensive Methods in Computer Vision*. 2006 (cit. on pp. 14–15).

[78] G. Toussaint. 'Solving Geometric Problems with the Rotating Calipers'. In: *Mediterranean Electrotechnical Conference*. 1983, pp. 10–17 (cit. on p. 23).

[79] S. Uchida. 'Text Localization and Recognition in Images and Video'. In: *Handbook of Document Image Processing and Recognition*. Ed. by D. Doermann and K. Tombre. Springer London, 2014, pp. 843–883 (cit. on p. 4).

[80] E. Wan and R. Van Der Merwe. 'The unscented Kalman filter for nonlinear estimation'. In: *Adaptive Systems for Signal Processing, Communications, and Control*. 2000, pp. 153–158 (cit. on pp. xiii, 25).

[81] K. Wang, B. Babenko and S. Belongie. 'End-to-end scene text recognition'. In: *IEEE International Conference on Computer Vision*. Nov. 2011, pp. 1457–1464 (cit. on pp. 5, 33).

[82] P. Wellner. 'Interacting with Paper on the DigitalDesk'. In: *Communications of the ACM* 36.7 (July 1993), pp. 87–96 (cit. on p. 3).

[83] C. Yao, X. Bai and W. Liu. 'A Unified Framework for Multioriented Text Detection and Recognition'. In: *IEEE Transactions on Image Processing* 23.11 (Nov. 2014), pp. 4737–4749 (cit. on p. 5).

[84] H. Zhang, K. Zhao, Y.-Z. Song and J. Guo. 'Text extraction from natural scene image: A survey'. In: *Neurocomputing* 122 (2013), pp. 310–323 (cit. on p. 4).

[85] J. Zhang and R. Kasturi. 'Extraction of Text Objects in Video Documents: Recent Progress'. In: *Document Analysis Systems*. 2008, pp. 5–17 (cit. on p. 4).