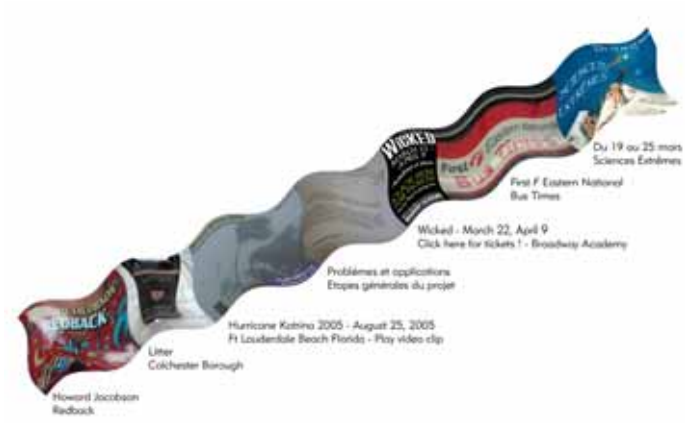
Natural Scene Text Understanding

CÉLINE MANCAS-THILLOU

2006

Natural Scene Text Understanding

Céline Mancas-Thillou



ACADÉMIE UNIVERSITAIRE
WALLONIE-BRUXELLES

POLYTECH.MONS

Registration of copyright: D/2007/9964/4

ISBN : 978-2-8746-3059-4
Cover : Colin MICHEL

Printed in Belgium.

Distribution : www.i6doc.com, on-line university publishers
This book and all others from the SIMILAR collection
are available on order from bookshops or at:

CIACO University Distributors
Grand-Place, 7
1348 Louvain-la-Neuve, Belgium
Tel. 32 10 47 33 78
Fax 32 10 45 73 50
duc@ciaco.com

Damien,
here is the relay baton.

Foreword

The work presented in this text has been realized in the framework of a PhD thesis and the text itself is actually the transcription of the thesis with some minor modifications. Although I wanted my PhD dissertation to be as clear as possible, it still sounds like a PhD dissertation in some places. Yet I hope that the introductive chapters (typically Chapter 1, 2, and 3) will be helpful to image processing or document analysis novices and the other ones will be readable (and motivating for further researches!) without too many efforts. It is the result of almost four years of researches spent mainly within the TCTS Lab of the Faculté Polytechnique de Mons in Belgium and with a delightful parenthesis within the Computer Vision Group of the University of Bristol in United Kingdom.

The thesis has been submitted to the Faculté Polytechnique de Mons for the degree of Doctor of Philosophy in Applied Sciences and has been publicly defended on December, 18^{th}, 2006. Members of the complete jury were:

- Prof. Pierre Manneback, Faculté Polytechnique de Mons, Belgium, Chair
- Prof. Franck Dubois, Université Libre of Bruxelles, Belgium
- Dr Laurence Likforman-Sulem, Ecole Nationale Supérieure des Télécommunications, France
- Prof. Paul Lybaert, Faculté Polytechnique de Mons, Belgium, Dean
- Dr Majid Mirmehdi, University of Bristol, United Kingdom
- Prof. Justus Piater, Université of Liège, Belgium
- Prof. Jacques Trécat, Faculté Polytechnique de Mons, Belgium
- Prof. Bernard Gosselin, Faculté Polytechnique de Mons, Belgium, Advisor

Abstract

Natural Scene Text Understanding

In a society driven by visual information and with the drastic expansion of low-priced cameras, text recognition is nowadays a fast changing field. In particular, natural scene text understanding aiming at extracting text from daily images is the main concern of this text.

From text extraction to correction of recognition errors, each sub-step is deeply studied to enhance versatility for handling most images, even the most complex ones.

Either in color camera-based images or in low resolution thumbnails, inherent degradations, such as complex backgrounds, artistic fonts, uneven lighting or unsatisfactory resolution, must be taken into account. In order to circumvent or correct them, studies of image formation and degradation sources challengingly led to overcome too constrained definitions of color spaces. Hence the selective metric text extraction attempts to combine magnitude and directional processing of colors in an unsupervised framework.

Text extraction from background is simultaneously linked to subsequent steps of character segmentation and recognition. This intermingled chain mainly aims at combining color, intensity and spatial information of pixels for robustness and accuracy. Each of these features addresses different issues; the first one for text extraction and the two latter ones for recovering initial separation between characters through log-Gabor filtering.

In order to reach higher quality results, pre- and post-processing of natural scene text understanding are necessary and deal with Teager-based super-resolution, assuming a simple affine motion between frames with the SURETEXT proposition for the first one and with association of recognition outputs and linguistic information through lightweight finite state machines for the second one.

In the final part of each step, results are clearly mentioned to highlight effectiveness of the methods. Moreover, several databases, to be independent of a particular one, and a public

and renowned data set, are used to assess results and compare them with recent and competing algorithms.

Finally a large discussion is opened through presented achievements of this text and required future extensions in natural scene text understanding to complete exciting applications, such as reading tool for visually impaired or innovative web images search engines in a life-log context!

Acknowledgement

I am writing this acknowledgement page at 1 AM in a hotel room during a conference as I could not sleep at all because of jet lag. It is the best time to remember all people who helped me during this PhD, with the calm of the night, far away from home.

First of all, I will start by thanking Prof. Bernard Gosselin, my PhD advisor, who trusted me during the past four some years, corrected my papers and this text, and regularly encouraged me with nice words. I also wish to thank the remaining academic staff, Prof. Joël Hancq and Prof. Thierry Dutoit, who provided a nice and pleasant atmosphere, made administrative annoyances easy and pushed towards interesting seminars and rigorous scientific works. A PhD cannot be completed alone, and even more so, must not be achieved by a single isolated person. It is useful to add new interesting contacts, papers, opinions and in my case to remotivate me to go towards a richer well-proven study.

[After my first PhD defense, I received nice remarks and very helpful corrections to make this text as perfect as possible. Hence, I wish to thank committee members who carefully read the manuscript and tracked each single error.]

During my PhD, at odd moments, some people appeared and were real boosts for my work. Hence, for a long time, I have really wished to thank Emmanuel Ademovic, one of my previous teachers during my first-master studies. With sparse words and an impressive background, he gave me some pieces of advice, web links, an interesting contact in Stéphane Chrétien, a doctor in mathematics, and nice conversations. Simply, he was curious about my work and it was worthwhile to meet people like him during my PhD. S. Chrétien, previously cited, gave me nice inputs for the Gaussian Mixture Modelling part of my work and also provided a really weird afternoon, by explaining stuff of his own works and its possible relationship to mine with electron spins...

Three more outstanding researchers really helped me during this PhD and this acknowledgement section is also dedicated to them. First off, Dr Majid Mirmehdi, who is a part of my thesis committee and with whom I electronically met during a negative answer for a too-early journal submission. He proposed for me to work in his laboratory at the university of Bristol for sev-

eral months. This experience was awesome with very hard work, which allowed us to complete Chapter 5 about super-resolution algorithms, write a workshop paper and a chapter in a book (special thanks for this opportunity). I thought I was thorough but after meeting Majid, I realized that I still need to provide much effort to reach his rigor. Second, Dr Justus Piater, who is also a member of my thesis committee, regularly pushed me forward with interesting ideas, and new ways of research. Unfortunately, I could not complete some of them for many reasons but they are in a small box in my mind. Maybe, one day, ... Third, Richard Beaufort who inspired me to link our collaborative works by adding linguistic information in recognition post-processing to increase final recognition rates. Richard helped me during my second-master works and during this last month to achieve comprehensive and very promising results of Chapter 8.

Finally, about these random contacts, I would like to thank Dan Popescu, a CSIRO researcher in Australia, with whom I met during ACIVS conferences for his obvious kindness. It is very nice to meet people like him who help you go forward with only a few words that stay with you for a long time. Similarly, by electronic contacts, I will thank Michael Hild, a Japanese researcher, who wrote me encouraging words, pieces of advice and provided me with papers without knowing me! It is also well deserved to thank researchers all over the world who make their papers and/or source code available on the Internet to speed up research!

In daily life, either FPMs or Multitel researchers, I wish to thank the Sypole team, which dynamically changed with Silvio, Vincent and Jonathan. The project Sypole is my main funding source by Région Wallonne of Belgium and allowed me to work with a team in everyday conditions on a end-project for blind or visually impaired people. For creation and maintenance of a friendly atmosphere, I thank all the TCTS laboratory and Multitel research center with special thanks for Sophie, Stéphanie, Pascale and Sylvie for the very useful feminine context; Raphaël, Vincent, Julien, Jean-François, Alain, Richard, Jo, Xavier R. and Xavier T. for their computational tricks, programming help and kindness. I also wish to mention Multivision group of Multitel for interesting seminars, exchanges and pleasant discussions with Jérôme, Caroline, Christophe, Xavier D., Bertrand, Derek, ... Finally, I thank Devrim, Raphaël and Matei for their parallel PhD work, their completion in almost in the same time as mine, nice discussions, and very helpful rereading (especially eagle-eyed De-

vrim, who read very carefully with in mind, his own thesis).

Speaking of rereading, I now mention Elias, an American and Lebanese friend that I met during an internship in Ohio, USA. It is very hard to ask somebody, external to your work, to correct your entire thesis with uncountable errors, but Elias did it without hesitation. He was even afraid that he was taking too much time to return the corrections to me! Thank you Elias! If this text contains obvious mistakes, it will be because of my rewriting, because I could not dare to ask him to reread it once again.

After completing this work in a university far from my childhood home, I gradually understood the superior-over-everything importance of family and friends in daily life. Hence, I will thank Rosalie, Valérie, Aziliz, Guillaume, Damien R., Evelyne, Florent, and Pauline for their nice parties, nice words, and nice refreshing air all through this work. It was once again proven in our last trip in Romania!

My parents and my brother, Damien, deserve all my thanks during this PhD and always. They always trust me and are proud of me with indefectible kindness. I am so happy when I go back home!

Finally, I will thank Matei, who became my husband during this PhD, for helping me every single day with nice conversations around our respective works and the help he gave me to realize Chapter 7 and papers along this work, his rereading of every line I wrote, and especially this text, his support during my stay in Bristol and in our daily life, and his relief and "cool" attitude against my fears, my troubles and my excessive stress. It is worthwhile to have somebody so close to me who is able to understand my work, answer my silly questions, and push me forward each time my motivation declines. I could not complete this text without Matei! Multumesc, dragul meu!

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ASCII	**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange
BIC	**B**ayesian **I**nformation **C**riterion
BST	**B**ackground **S**urface **T**hresholding
CIE	**C**ommission **I**nternationale de l'**E**clairage
CIF	**C**ommon **I**ntermediate **F**ormat
CMOS	**C**omplementary **M**etal **O**xide **S**emiconductor
CMY	**C**yan, **M**agenta, **Y**ellow color space
CMYK	**C**yan, **M**agenta, **Y**ellow and **K**ey color space
Dpi	**D**ots **p**er **i**nch
DRM	**D**ichromatic **R**eflection **M**odel
EM	**E**xpectation-**M**aximization
FJ	**F**igueiredo and **J**ain method
FSA	**F**inite **S**tate **A**utomaton
FSM	**F**inite **S**tate **M**achine
FST	**F**inite **S**tate **T**ransducer
GHz	**G**iga **H**ertz
GMM	**G**aussian **M**ixture **M**odelling
HID	**H**andheld **I**maging **D**evice
HMM	**H**idden **M**arkov **M**odel
HP	**H**ewlett-**P**ackard
HR	**H**igh-**R**esolution
HSI	**H**ue, **S**aturation and **I**ntensity color space
HSV	**H**ue, **S**aturation and **V**alue color space
HTC	**H**otelling **T**race **C**riterion
IBP	**I**terative **B**ack-**P**rojection
ICDAR	**I**nternational **C**onference on **D**ocument **A**nalysis and **R**ecognition
JPEG	**J**oint **P**icture **E**xperts **G**roup
k**-NN**	k-**N**earest **N**eighbor
Lab	CIE 1976 **L*****a*****b*** color space (L for Lightness, a, b for red/blue and yellow/blue chrominances)
Lch	CIE $\mathbf{L}^*\mathbf{CH}^\circ$ color space

	(L for Luminance, C for Chroma and H for Hue)
LR	**L**ow-**R**esolution
Luv	CIE 1976 **L*****u*****v*** color space
	(L for Lightness, u, v for chrominances)
MAP	**M**aximum **A** **P**osteriori
MDSP	**M**ulti-**D**imensional **S**ignal **P**rocessing
MIMO	**M**ultiple **I**nput **M**ultiple **O**utput
MISO	**M**ultiple **I**nput **S**ingle **O**utput
MLP	**M**ulti-**L**ayer **P**erceptron
MML	**M**inimum **M**essage **L**ength
MPEG	**M**oving **P**icture **E**xperts **G**roup
MRF	**M**arkov **R**andom **F**ield
***n*D**	***n*-D**imensional (n is an integer)
NLP	**N**atural **L**anguage **P**rocessing
NS	**N**atural **S**cene
OCR	**O**ptical **C**haracter **R**ecognition
OOV	**O**ut-**O**f-**V**ocabulary words
PDA	**P**ersonal **D**igital **A**ssitant
PDF	**P**robability **D**ensity **F**unction
POCS	**P**rojection **O**nto **C**onvex **S**ets
PSF	**P**oint **S**pread **F**unction
QVGA	**Q**uarter/**Q**uick **V**ideo **G**raphics **A**rray
RANSAC	**RAN**dom **SA**mple **C**onsensus
RGB	**R**ed, **G**reen and **B**lue color space
ROC	**R**eceiver **O**perating **C**haracteristics
SCT	**S**pherical **C**oordinate **T**ransform
SISO	**S**ingle **I**nput **S**ingle **O**utput
SMC	**S**elective **M**etric **C**lustering
SOM	**S**elf-**O**rganizing **M**aps
SR	**S**uper-**R**esolution
SSD	**S**um of **S**quare **D**ifference
SURETEXT	**SU**per-**R**esolution **E**nhanced **TEXT**
TTS	**T**ext-**T**o-**S**peech
TV	**T**otal **V**ariation
UML	**U**nified **M**odelling **L**anguage
VA	**V**iterbi **A**lgorithm
WFSA	**W**eighted **F**inite **S**tate **A**utomaton
WFSM	**W**eighted **F**inite **S**tate **M**achine
WFST	**W**eighted **F**inite **S**tate **T**ransducer
WWW	**W**orld **W**ide **W**eb
XML	E**X**tensible **M**arkup **L**anguage
XYZ	CIE 1931 **XYZ** color space

YCbCr	Luminance(**Y**), **C**hromaticity **b**lue and **C**hromaticity **r**ed color space
YIQ	Luminance (**Y**) **I**n-phase **Q**uadrature color space
YUV	Luminance (**Y**) and Chrominance (**U,V**) color space

— CHAPTER 1 —

Introduction

1.1 Current Document Analysis

Inputs for traditional document analysis systems use scanner-based acquisition with flatbed, sheet-fed or mounted imaging devices. Recently, handheld scanners such as pen-scanners appeared to acquire small parts of text on a fairly planar surface such as that of a business card. Issues having an impact on image processing are limited to sensor noise, skewed documents and inherent degradations to the document itself. Based on this classical acquisition method, optical character recognition (OCR) systems have been designed for many years to reach a high level of recognition with constrained documents, meaning those falling into traditional layout, with relatively clean backgrounds such as regular letters, forms, faxes, checks and so on and with a sufficient resolution (at least 300 dots per inch (dpi)).

Research on scanned documents obviously focuses on the document, attempting to handle various fonts, curled pages, text and graphic separation, handwritten recognition and so on. With the recent explosion of handheld imaging devices (HIDs), i.e digital cameras, standalone or embedded in cellular phones or personal digital assistants (PDAs), research on document image analysis entered a new era where breakthroughs are required: traditional document analysis systems fail against this new and promising acquisition mode and main differences and reasons of failures will be detailed in this introductive chapter. Small, light, and handy, these devices enable the removal of all constraints and all objects,

such as scenes in different situations in streets, at home or in planes may be now acquired! Moreover, recent studies [65] announced a decline in scanner sales while projecting that sales of HIDs will keep increasing over the next 10 years. Much effort must be brought to prompt expected and promising applications. A first workshop on camera-based document analysis and recognition[1] even appeared in 2005, in order to focus research on natural scene image analysis.

1.2 What is Natural Scene Text?

Several definitions are given to identical terms in literature creating confusion regarding their meaning. In order to understand the challenges of natural scene (NS) text understanding, it may help to clear up differences between synthetic or real images, still images or video frames, caption, scene text or camera-based documents. Definitions and types of images (Figure 1.1) are listed:

Synthetic versus real images: Properly named, synthetic images are designed by computers to reproduce real events or degradations. The main aim is to increase database size and experiment with new algorithms. Unfortunately, it is rather difficult to combine and even identify all degradations coming from real images acquired by recent HIDs.

Still images versus video frames: In video frames, temporal information redundancy, depending on frame rate, brings additional data enabling statistical methods to work more efficiently. Still images are snapshots of a scene providing only their own information for further analysis.

Caption, scene text and camera-based document text: Caption text is artificial text superimposed on an image or a video frame such as subtitles or scores in a tennis game. It is therefore not correlated to degradations present in the scene. In contrast to that, scene text and camera-based document text are integral parts of the picture, such as labels on a bottle, street names and so on. Camera-based documents are the same documents usually acquired by a scanner and presenting a particular layout with titles or paragraphs; those are usually absent in scene text. Research

[1] http://www.m.cs.osakafu-u.ac.jp/cbdar

on camera-based documents focuses more on recognition, unwarping, layout analysis. Moreover entire document images are hardly acquired in one image due to the low resolution of popular HIDs and the required resolution for OCR. It implies mosaicing techniques to recompose the document based on several contiguous snapshots or frames of a video sequence. It is not the case with scene text where writing density is lower.

This text will mainly consider scene text, parts of real still images coming from various HIDs and some web images presenting partially similar issues. Only typewritten text will be processed; handwritten text in a camera-based document or a natural scene is analyzed with different methods based on handwriting recognition. Conventional document analysis techniques perform very poorly on scene text due to the new imaging conditions and newly considered scenes:

1. ***Imaging Conditions***

   Raw sensor image and sensor noise: In low-priced HIDs, pixels of a raw sensor are interpolated to produce real colors, which can induce degradations. Demosaicing techniques, viewed more as complex interpolation techniques, are sometimes required. Moreover, sensor noise of an HID is usually higher than that of a scanner.

   Viewing angle: Scene text and HIDs are not necessarily parallel creating perspective to correct.

   Motion Blur: During acquisition, some motion blur can appear or be created by a moving object. All other kinds of blur are included in some other imaging conditions such as focus, for example.

   Focus: HIDs are not necessarily equipped with auto-focus and lens aberration can strongly blur the image.

   Lighting: In real images, real (uneven) lighting, shadowing, reflections onto objects, inter-reflections between objects may make colors vary drastically and decrease analysis performance.

   Resolution and Aliasing: From webcam to professional cameras, resolution range is large and images with low

resolution must also be taken into account. Resolution may be below 50 dpi which causes commercial OCR to fail. It may lead to aliasing creating fringed artifacts in the image. It is also the case for WWW (World Wide Web) images used for fast internet transmission.

2. ***Considered Scenes***

Outdoor/non-paper objects: Different materials cause different surface reflections leading to various degradations and creating inter-reflections between objects.

Scene text: Backgrounds are not necessarily clean and white, and more complex ones make text extraction from background difficult. Moreover scene text such as that seen in advertisements may include artistic fonts. Due to the large diversity of backgrounds and text to handle, it is rather difficult to detect and extract text in the image.

Non-planar objects: Text embedded in bottles or cans suffer from deformation.

Unknown layout: There is no a priori information on structure of text to detect it efficiently.

Objects in distance: Distance between text and HIDs can vary, and character sizes may vary in a wide range, leading to a wide range of character sizes in a same scene.

Moving objects: A moving camera (by its mobile context) or moving objects may induce unknown motion blur, which is difficult to model, leading to degraded imaging conditions, previously detailed.

These definitions are illustrated by some examples in Figure 1.1 and lead to the explanation of NS text images:

Natural Scene Text: Textual part of still images or video frames of a scene with no a priori knowledge of environment, lighting, objects supporting text, acquisition parameters and finally text itself. It could easily be viewed as text in real-world conditions without any constraints or assumptions.

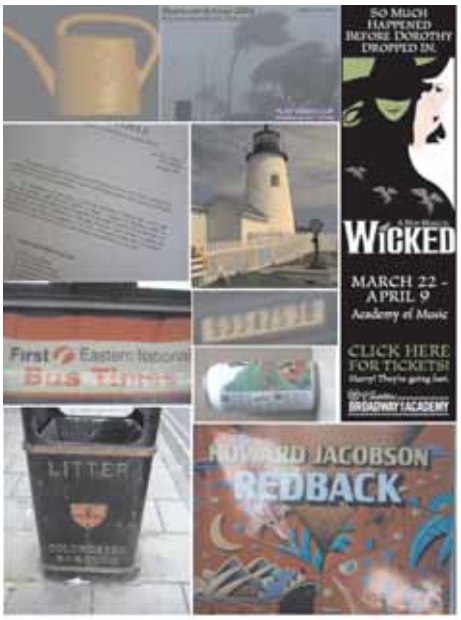


Figure 1.1: Samples of images of different kinds considered or not in the text. From left to right and top to bottom: a synthetic image, a video frame with caption text, a web image, a camera-based document, a real image with raw sensor degradation, a scene text with uneven lighting, a license plate with blur and low-resolution, text embedded in a non-planar object, a scene image with different character sizes, a scene image with complex backgrounds and varying colors.

NS text differs from scanner-based document text by the combination of degradations and numerous unknowns to solve individually or not. Figure 1.2 shows the area where scientific community has to contribute to make applications described in the next section more accessible.

1.3 Numerous Applications

As HIDs become more and more powerful, on-the-fly image processing becomes possible, opening up a new range of applications. Nevertheless, today's HIDs are easily connected to various net-

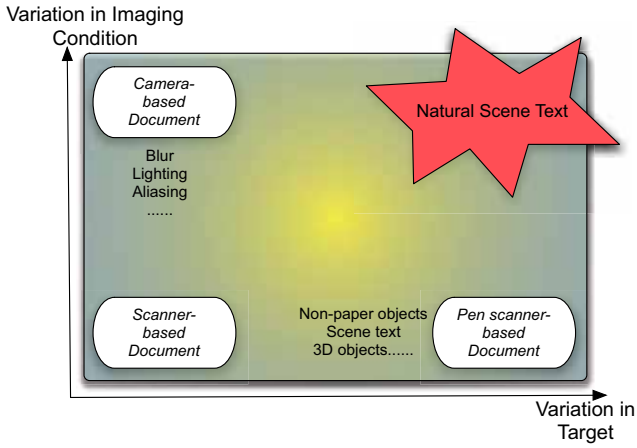


Figure 1.2: Several levels of difficulties for text analysis: NS text understanding must deal with variations in imaging conditions and in targets.

works and supplementary computing resources. Starting from sign recognition for foreigners for the 2008 Olympic Games in Beijing, navigation with visual tags for surveillance robots, automatic license plate recognition to driver assisted systems with text projection on windshields, various situations could be handled. A more useful application for researchers themselves is poster or slide capturing during a conference... Interesting applications such as mobile phones operating as fax machines even led to strict sanctions in Japanese bookstores!

In March 2006, a new company, entitled 'Scene Reader'[2], was launched, whose aim is to recognize NS text in different situations with a dedicated software. The proposed algorithms of this text may improve results. Comparatively, a visual search engine is born is 2005. Its name is Riya[3] and enables the search of similar images by directly uploading images or the search of pictures containing the given text, among many other kinds of searches. Results are really impressive!

The well-known DjVu format, created by Bottou et al. [10],

[2]http://www.scenereader.com/
[3]http://www.riya.com/

compresses digitized images with different algorithms for textual foreground and background and performs poorly on scene text embedded in background, leading to low quality. NS text understanding may help to properly extract text from backgrounds to apply a particular compression to scene text, as already forecast in DjVu. In the same way, the MPEG-7 standard provides information to index video sequences, by using image segmentation to identify objects, people but also caption text. Coding models are no more pixel-based but object-based and NS text understanding may extract text from the scene to give additional information for content-based retrieval.

Visually impaired people are directly affected by such research and were the initial motivation for this work [34, 44, 45, 117]. With an HID and sufficient resources, scene in daily life may be analyzed to give them access to text and, coupled with a text-to-speech algorithm, make them "read" book covers, banknotes, labels on office doors, medicine labels and so on. For the blind community, such devices are really expected, proven by the recent launch of BlindReader[4] of Kurzweil Technologies! This reading assistant assumes well-contrasted text on documents and is not currently designed for NS text.

Applications are very numerous and currently only limited by imagination. Applications such as "life-log" may one day be possible with NS text understanding.

1.4 Text Understanding System: Main Steps

How does one achieve the pre-cited applications? By using *a text understanding system*. We prefer this last term to *text segmentation* which have several definitions in literature such as *text detection*, *text binarisation/extraction* or *segmentation of text into individual components* (what we will refer to *character segmentation*). *Text segmentation* is invariably used for each of these steps, which is inconvenient.

A text understanding system, described in Figure 1.3, encompasses three main steps: text detection and localization, text extraction from background, and text recognition.

Text detection and localization: This field finds answers to

[4] http://www.knfbreader.com/

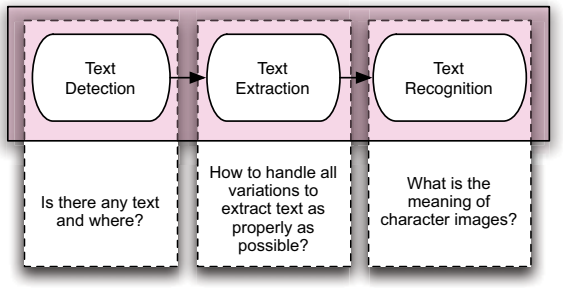


Figure 1.3: An overall text understanding system.

the question: "Is there any text and where is it?". This part is quite difficult in NS images and has been extensively studied during previous years. The reader may refer to the excellent survey of Jung et al. [59].

Text extraction from background: Also named "text extraction" or "text segmentation", it is the field dealing mainly with uneven lighting and complex backgrounds. It is a paramount step to prepare data for OCR. Classical image segmentation such as separating sky from mountains does not need as much accuracy as text extraction, which is considered more as object-driven segmentation. Actually, text is a meaningful object which has to be extracted properly to be better recognized afterwards.

Text recognition: This is the final step to convert character images into ASCII values to understand text and use it for particular applications.

Other NS text analysis steps such as warping, mosaicing or text tracking are also part of text understanding systems for different applications and for more details, the reader may refer to the overall state-of-the-art of Liang et al. [75].

1.5 Challenges and Overview of Problem Bounds

The main challenge is to design a system as versatile as possible to handle all variability in daily life, meaning variable targets with unknown layout, scene text, several character fonts and sizes and variability in imaging conditions with uneven lighting, shadowing and aliasing. Proposed solutions for each text understanding step must be context independent, meaning independent of scenes, colors, lighting and all various conditions. Nevertheless, structural context based on neighborhood pixels or neighboring steps is a rich source of information and needs to be exploited very efficiently. Hence we focus on methods which work reliably across the broadest possible range of NS images.

It is rather difficult as all degradations could not be corrected individually because of the high interdependency between some of them. Moreover, several degradations are irreversible such as illumination, aliasing or blur. They induce ill-posed problems, which have no unique solution by definition.

Particular focus is cast on the text extraction step: it is declared as the "most important factor for high performance" by In-Jung Kim [65], a senior researcher at Inzisoft, which recently launched *Mobile Reader*^{TM}, a software reading application for smartphones. Slightly studied since the inception of camera-based text analysis, text extraction suffers from imaging conditions and, based on a thorough study on text extraction itself, low resolution problem is also taken into account. On the other hand, the text detection step will be only briefly mentioned in this text. S.Lucas, after the ICDAR (International Conference on Document Analysis and Recognition) 2005 text locating competition [1], was able to conclude that "in text locating, [...] there has been a significant advance in performance [and] most easy-to-read (for humans) text is now well detected". He also mentioned that variations in illumination such as reflections cause significant problems for text understanding. Hence, considerations on uneven lighting and how to circumvent it for efficient text extraction are particularly highlighted as well.

1.6 Overall Structure

This text falls naturally into two parts, the first one dealing with background and overview of the whole system, and the second one with some solutions for main text understanding challenges.

These two parts are decomposed into nine chapters, including this one:

Chapter 2 describes background information about image formation where image quality is influenced by light on image quality. The triplet Light, Object and Camera is considered to understand image formation. In the second part, several color spaces are described with their advantages and disadvantages.

Chapter 3 with the previous chapter ends the description of background on text extraction and additional steps to achieve an efficient text understanding system such as resolution improvement or character segmentation. Literature survey is also browsed along these lines.

Chapter 4 merely deals with the overview of the whole system by detailing the proposed feedback-based chain. Material and databases used for experiments are also sketched out.

Chapter 5 encompasses pre-processing steps for an efficient text extraction algorithm and describes resolution enhancement techniques for both still images and video sequences. It highlights one of the thesis contributions with the SURETEXT algorithm using several frames to get a higher-resolution image and assuming a simplified affine motion between frames.

Chapter 6 forms the main body of the text with the selective metric clustering (SMC) algorithm for text extraction. The proposed solution is detailed with justifications of each step and several experiments including comparisons with other recent techniques to highlight the performance of the whole method. Instead of using several color spaces to segment text from background, several agglomerative metrics for color pixels are preferred based on image formation. To perfectly complete this step, the following step of character segmentation is used to add robustness to the solution.

Chapter 7 is devoted to segmentation of extracted text into individual units such as characters to improve recognition afterwards. Log-Gabor filters, well designed for NS images, are used here for the first time for character segmentation into individual components and represent the third contribution of this text. Parameters of the filter are automatically tuned with the following step of character recognition. Robustness against italic characters, strongly joined and broken characters is also addressed. Segmentations into line and words for text understanding are mentioned in this chapter as well.

Chapter 8 can be viewed as considerations on character recognition and correction to get an understandable text, especially to feed into a text-to-speech algorithm. Optimization of character recognition is outside the scope of this text but some clues such as representativeness of the database or character magnification are detailed. Correction of recognition is a necessary step to get an exploitable recognition for applications. The previous solution based on bi- or trigram history using linguistic information and the Viterbi algorithm to find the most relevant path is described and compared to the fourth contribution of this text: the combination of linguistic information, easily modelled by finite state machines and the output of character recognition, not considered here as a closed "black box".

Chapter 9 ends this text with conclusions about text understanding for NS images and remaining issues. Contributions are clearly mentioned before discussing the future of this work.

— CHAPTER 2 —

Image Formation and Representation

To deeply understand challenges of this text, this chapter describes firstly all sources of degradations in natural scene images. Image representation, through color spaces, is then detailed to mention advantages and drawbacks of some spaces which guided research.

2.1 Image Formation: Why do Colors Vary for the same Object?

Perceived illumination can vary drastically depending on the surrounding environment and these changes induce varying perceived colors. One of the human mechanisms for color consistency is chromatic adaptation, based on a chromatic behavior. In color segmentation, research attempts to reproduce the same effect for computers, which means merging similar colors independent of viewing conditions and environment. Hence, the trio of, light, object and camera, must be considered to evaluate all possible degradations and color variations.

2.1.1 Light

Light can be emitted, absorbed, or reflected by a surface, or simply pass through it. Emission, transmission and absorption are out of scope of this text as they do not influence colors or if they do, the

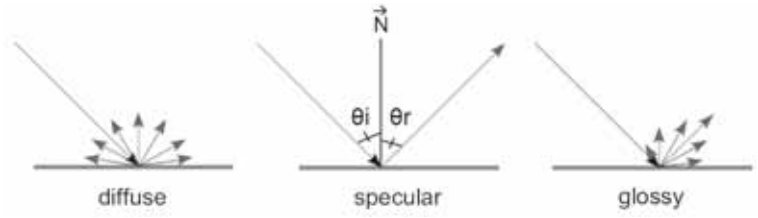


Figure 2.1: Different reflection types: diffuse, specular and glossy.

influence is minimal, for instance, for transmission surfaces such as glasses.

Except for pictures of direct light such as photographing the sun, colors of a scene are viewed with indirect illumination, due to reflected light onto objects. *Reflection* is therefore the proportion of any incident light arriving at a surface and reflected back into the environment.

To understand all causes of color variation in a scene, it is convenient to study existing models of light reflection. As stated in [133], natural light has a diffuse behavior in which rays do not have a constrained orientation. Nevertheless in NS images, directional lights sources such as flashes or spots must also be considered. In further explanations, light is modelled by one ray coming either from a diffuse or directional light.

2.1.2 Object

Dielectric objects, meaning nonhomogeneous material, are the most common objects and can induce different reflection types as illustrated in Figure 2.1:

Diffuse reflection means reflection in all directions equally while **specular surfaces** reflect light in a particular direction with the angle of incidence, θ_i, equal to the angle of reflection, θ_r, following the laws of optical reflection without diffusion as exemplified by mirrors. **Glossy reflection** concerns combination of diffuse and specular ones to handle natural surface irregularities: light is reflected in every direction but more in a constrained one which is the reflection direction of specular surfaces. In reality, all surfaces are more or less glossy [162].

To model glossy objects, the Dichromatic Reflection Model (DRM), introduced by Shafer [132], states that light is reflected on dielectric materials in diffuse and specular reflection. Light I_r reflected from a colored object surface is a function of pixel location x and light wavelength λ:

$$I_r = \text{diffuse reflection} + \text{specular reflection} \tag{2.1}$$

$$I_r = \alpha_d(x)S(\lambda)E(\lambda) + \beta_s(x)E(\lambda) \tag{2.2}$$

where $E(\lambda)$ is the spectral power distribution of a light source, $S(\lambda)$ is the spectral-surface reflectance of an object, $\alpha_d(x)$ is the shading factor and $\beta_s(x)$ is a coefficient for the specular reflection term.

Two of the most simple, but representative models, are the Lambert [4] and Phong [118] models which detail respectively, diffuse and glossy reflection (sometimes abusively called specular reflection).

Diffuse reflection: matte surfaces or Lambertian reflectors are considered in this case under the assumption of a white light source. The distribution of exiting light can be described by Lambert's cosine law, which states that the reflected light I_r appears equally bright regardless of viewing conditions. Light perceived I_p by the camera or the observer, which is equal to I_r, is the product of intensity of the light source I_s by the cosine of the angle θ_i between I_s and the normal direction $\overrightarrow{N}$ to the surface (Equation 2.3), perturbed by the shading factor, $\alpha(x)$. Hence, as the θ_i increases, the amount of light decreases.

$$I_p = I_s \times cos(\theta_i) \times \alpha(x) \tag{2.3}$$

Gevers [46] concluded that a uniform colored surface which is curved returns different intensity values to the camera. Figure 2.2 displays the Lambert's cosine law for different locations of a curved surface. This case is quite frequent in NS images.

Glossy reflection: this case refers to shiny objects, presenting a globally symmetric reflection to the normal direction $\overrightarrow{N}$, hence with reflected intensity I_r depending on the viewing conditions. Phong's model [118] describes the geometry of image formation for computer generated images and eases the understanding of color variations in an image. The camera's viewing

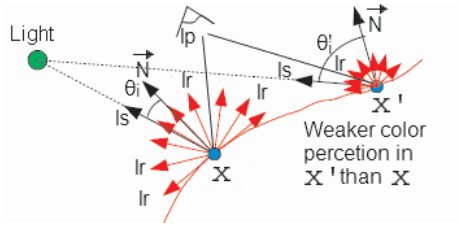


Figure 2.2: Difference between diffuse reflection in different locations on a curved surface. The observer perceived different intensity values.

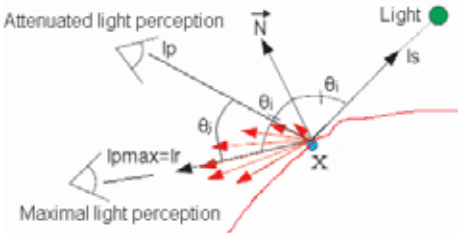


Figure 2.3: Glossy reflection defined by Phong's model. Perceived intensity is maximum (Ip_{max}) when $I_p = I_r$.

angle is fixed in NS text extraction but color varies with the surface orientation as well, leading to highlights. Figure 2.3 shows the orientation of the exiting surface reflection I_r. The perceived intensity I_p is a function of the angle θ_j between I_p and I_r, described by Equation 2.4.

$$I_p = I_s \times \cos^n(\theta_j) \times \beta_g(x) \tag{2.4}$$

where $\beta_g(x)$ defines the glossy coefficient for the point x and n is the diffusion coefficient around I_r, attenuating the perceived light when I_p is different from I_r.

Moreover, Phong's model is quite representative of the "flash" effect on glossy surfaces as described in [120]. This phenomenon is recurrent in NS images acquired by a camera and a flash or containing shiny surfaces.

Interreflection between objects is a generalization of previous cases using a single light source because reflections onto objects

are considered as another light source. Shadows are present due to obstacles (other objects) between light and object to be viewed. Light in the shadowed part results from other attenuated parts of incident light around.

2.1.3 Camera

The camera sensor may be viewed as the observer's eye and different viewing angles induce different color perception of an object. An image taken with a linear device such as a digital color camera is composed of sensor responses that can be described by Equation 2.5:

$$p = \int_w C(\lambda)Q(\lambda)d\lambda \tag{2.5}$$

where p is a 3-vector of sensor responses leading to pixel values, λ is the wavelength of light, C is the color signal (the light reflected from an object), and Q is the 3-vector of sensitivity functions of the device. Integration is performed over the visible spectrum w (380-780 nm). The 3-vector to represent colors is born from studying the human visual system with the 3 types of the eye's photoreceptors and Maxwell's color matching experiments [96] to reproduce all colors. It corresponds to Red, Green and Blue values leading to the well-known RGB color space.

All viewing conditions, matte or shiny surfaces and diffuse or directional illumination source induce that:

- Two identical (or different) colors in an object may be perceived identically (or differently) by the camera, which is the usual case with almost no degradations. This case occurs with matte and plane surfaces for example.
- Two different colors in an object may be perceived identically by the camera. This phenomenon is called illuminant metamerism where two colors match when viewed under one light source, but do not match when viewed under another, and vice versa.
- Two identical colors in an object may be perceived (slightly) differently due to a curved matte or shiny surface for exam-

ple. This case is the main issue of object-driven segmentation where similar colors must be merged even with (slightly) different perceived colors by the camera.

The color formation in a camera sensor, explained in this section, does not handle all unknown sources of variations, present in NS images but emphasizes the complexity of image formation and the mandatory challenge to handle varying colors in a scene.

2.2 Image Representation: Why do Different Color Spaces Exist?

A color space is represented by a multidimensional vector and pictures acquired by digital cameras use the most popular one, known as RGB, the physical sensor-based color space. Nevertheless, various other color spaces have been designed for different purposes.

In 1931, the "Commission Internationale de l'Éclairage" (CIE) adopted different norms to create standard color spaces to use. To define color spaces for devices, as explained in Section 2.1, light and observer are of the utmost importance. Hypothetical standard observers have been set to a viewing angle of 2° or 10°, inducing that each standardized color space is defined twice, for each observer. In regards to light, most color spaces are defined with respect to a white point, being the color reproduced by equal red, green and blue components. To obtain it, several illuminants are established and for the daylight, it is the CIE illuminant D65. Other illuminants, for printing for example, exist but we only consider D65 in the subsequent explanation.

In color segmentation, and more specifically in text extraction, independence on changing illumination intensities is expected. The normalized RGB (rgb) is invariant to illumination variation and defined by:

$$r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B}, b = \frac{B}{R+G+B} \tag{2.6}$$

Nevertheless, it is very noisy at low intensities due to nonlinear transformation.

RGB space excludes a few visible colors and the CIE defined the CIE XYZ tristimulus by a linear transformation to solve this

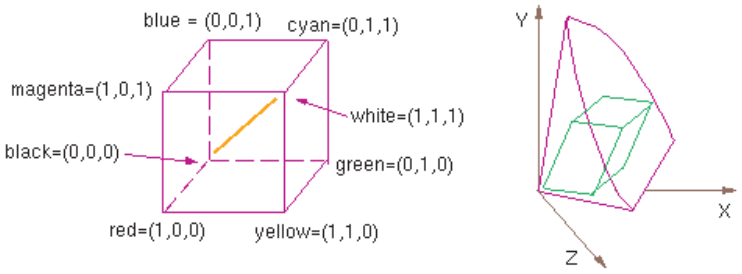


Figure 2.4: Display of the RGB cube alone (left) and inside the XYZ color space (right) illustrating the location of visible colors in XYZ.

problem (cf. Appendix A). For further explanations, its name will be simplified by XYZ. Nevertheless, a proportion of the XYZ domain does not correspond to colors visible by humans, contrarily to RGB. Figure 2.4 shows the representation of the RGB cube inside the XYZ color space. Y corresponds to luminance information, what is seen on gray-scale images. To merge similar color pixels together, a distance is usually determined and has to be representative of what is actually perceived. For that purpose, the MacAdam ellipses [162], shown in Figure 2.5 in the normalized XYZ color space (xyz), are such that colors inside the ellipses' boundaries are indistinguishable by the human eye. It is easily noticeable that using only the well-known Euclidean distance D_{eucl} between two colors defined by:

$$D_{eucl}(color1, color2) = \sqrt{\sum_{i=1}^{3}(color2_i - color1_i)^2} \qquad (2.7)$$

leads to non-realistic distances. In the bottom left corner of Figure 2.5, a small Euclidean distance represents large differences of colors, whereas in the upper left corner, a large Euclidean distance leads to small differences of colors.

MacAdam ellipses demonstrated that differences in XYZ are a poor guide to differences in color as colors are better represented with a context of magnitude (values) and orientation (ellipse's major axis). The same conclusion appears in the RGB color space due to the linear transformation between both color spaces.

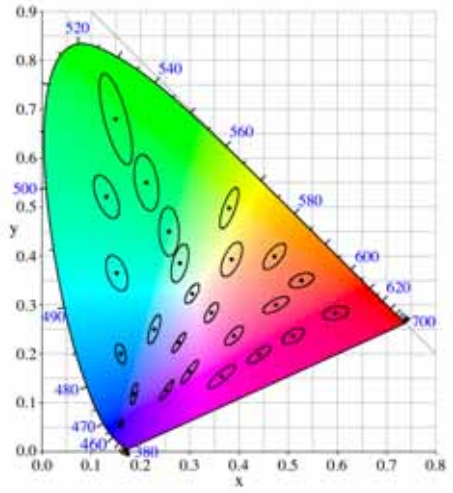


Figure 2.5: MacAdam ellipses denote that colors inside an ellipse's boundaries are indistinguishable by a human observer.

CIE L*a*b* and CIE L*u*v* (simplified respectively by Lab and Luv) are perceptually uniform color spaces, meaning that a small (and large) Euclidean distance between two colors leads to a small (and large) perceived color difference. They are converted from RGB by a non-linear transformation, recreating the logarithmic response of the human visual system. These perceptual color spaces give good results for image segmentation, as studied in a survey of Skarbek and Koschan [136]. The L^*CH° (Lightness, Chroma and Hue) color space is equivalent to Lab, however the color is located using cylindrical coordinates. It shall be simplified by Lch. Roughly speaking, lightness is the perceptual response to luminance, which is equal to Y in XYZ, chroma is the purity of a color also called saturation and hue is the dominant wavelength of a color. These two latter concepts describe together chromaticity of a color. Figure 2.6 illustrates these concepts. Many CIE system users prefer the Lch method of specifying a color, since the concept of hue and saturation agrees with the visual experience. These notions are actually more understandable by users.

Another type of color spaces highlights hue, saturation and luminance of a color such as HSI and HSV, which stand for Hue, Saturation and Intensity or Value, respectively. Hue is consid-

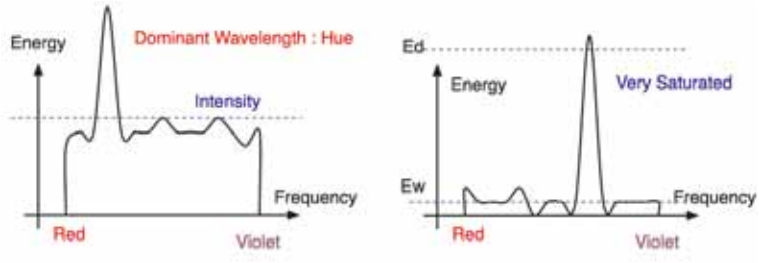


Figure 2.6: Illustration of intensity (lightness) and hue on left and saturation on right; Lightness is the area under the curve while saturation is equal to $Ed - Ew$ where Ed is the energy density of the dominant light and Ew is the contribution of the other frequencies producing white light.

ered as a good representation of colors, since different colors in MacAdam ellipses in XYZ or RGB have the same hue. Hence, the distance between two hues of colors inside one ellipse will be zero, which is required for image segmentation. Hue is considered as invariant to certain types of highlights and shadows which is expected in NS text understanding as well.

Statistical color spaces were recently used to circumvent rigid CIE definitions with a predefined observer and illuminant. The most general one is $X_1X_2X_3$ determined dynamically with a principal component analysis of image colors, leading to computation of eigenvectors, forming the basis for a new decorrelated color space. For more details, see the Karhunen-Loeve transformation [139]. In order to limit computation time, a fixed approximation of $X_1X_2X_3$ created the $I_1I_2I_3$ [112] color space. Actually, the first eigenvector of an image quasi-represents the luminance, whereas the second and third ones represent a linear combination of R, G and B components highlighting properties of the human visual system. I_1 has a low-frequency cutoff to be insensitive to average luminance while I_2 and I_3 are sensitive to absolute chromaticity.

The number of existing color spaces keeps increasing and it is not relevant to mention them all. Nevertheless, other color spaces have been designed for particular applications such as printing, which uses the CMYK model (Cyan, Magenta, Yellow, Key (black)), or television color systems using YIQ (Luminance (Y) In-phase Quadrature) and YUV (Luminance (Y) and Chromi-

nance (U,V)), or more generally video systems, which is based upon YCbCr (Luminance, Chrominance blue and Chrominance red) but these are out of scope of this text.

Conversions between various color spaces defined in this section are detailed in Appendix A.

2.3 To Summarize...

- The number of causes for color variation is large and trying to correct each issue independently is a utopia. Moreover some causes combined together may create other problems. Hence the variation of colors must be taken into account as an input to a text understanding system and solutions have to be found in order to handle them.
- Different colors (chromaticities) of an object may be identical while luminance information may be different or vice and versa. A solution using luminance information and chromaticities independently will be expected, which is rendered by an algorithm using color and gray-scale values.
- Due to MacAdam ellipses computation, colors have a magnitude and an orientation to consider in a color space.
- It is interesting to understand how color representation and distance evaluation between pixels interfere with text extraction results.
- Color spaces are historically built assuming a standard observer and standard lighting conditions, which is not versatile at all to handle the diversity of NS images. A solution which would not be based on conventional color space properties will be interesting as well.
- Finally, all color spaces are derived from the RGB space, which implies computationally expensive conversions. As such, a compromise between quality of text extraction and computational cost needs to be considered.

— CHAPTER 3 —

Background and Literature Survey of Text Understanding

Text understanding systems include three main topics: text detection, text extraction and text recognition. As explained in Chapter 1, we assume images input into the system have previously detected text if there is any in the image. Most papers describe independent methods for each of these three tasks and the system has been tested against accuracy of the text detection part. A text extraction system usually assumes that text is the major input contributor, but also has to be robust against variations in the detected text's bounding box size. For a detailed survey on text localization methods, usually grouped into region-based, edge-based, connected components-based and texture based, the reader may refer to the survey of Jung et al. [59]. Hence, this chapter details state-of-the-art methods of text extraction in the first section and, in the second section, discusses the usual additional steps taken to improve text extraction.

3.1 State-of-the-Art of Text Extraction

Text extraction is a critical and essential step as it sets up the quality of the final recognition result. It aims at segmenting text from background, meaning isolated text pixels from those of background. A very efficient text extraction method could enable the use of commercial OCR without any other modifications. Hence, we focus on this step in this chapter to better understand where the system, explained in the coming chapters, fits in.

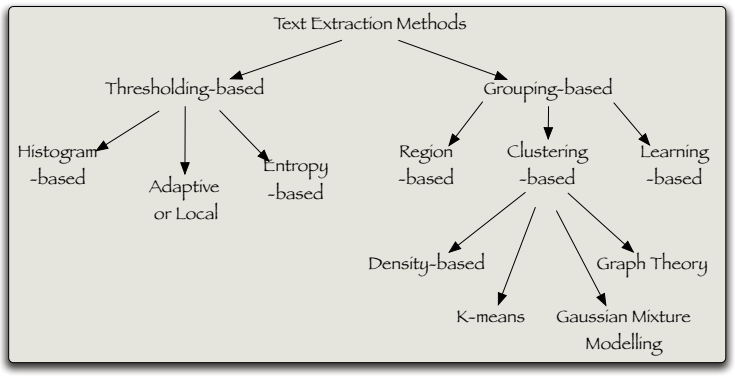


Figure 3.1: Classification of text extraction methods.

Due to the recent launch of the NS text understanding field, initial works focused on text detection and localization and the first NS text extraction algorithms were computed on clean backgrounds in the gray-scale domain. Following that, more complex backgrounds were handled using color information. Identical binarization methods were at first used on each color channel of a predefined color space without real efficiency for complex backgrounds, and then more sophisticated approaches using 3D color information, such as clustering, were considered. The classification of text extraction methods is displayed in Figure 3.1 and will be detailed further. Some issues of NS images are present in color camera-based documents for uneven lighting or low resolution, or in non classical documents, such magazines, historical papers for complex backgrounds, artistic character fonts and sizes, for example. Hence, some references of these applications will be mentioned as well.

3.1.1 Thresholding-based methods

Thresholding-based methods, as the name implies, define a threshold globally (for the whole image) or locally (for some given regions) to separate text from background. For example, pixels below the threshold will be set at 0 (for black) and pixels above, at 255 (for white).

- **Histogram-based thresholding** is one of the most widely used techniques for monochrome image segmentation. Images are composed of several homogeneous regions with different pixel values; text is one of these regions. A histogram counts the number of each pixel value from 0 to 255 in the image. Peaks (or modes) in histogram (meaning that several pixels have this same value) are considered as regions to segment. The threshold is chosen as the value corresponding to the valley between two peaks.

  The most referenced method is the one described by Otsu [113], which minimizes the weighted sum of within-class variances of the foreground and background pixels to get an optimum threshold as in [30, 143] for a visually impaired-driven application. Messelodi and Modena [97] chose two thresholds to strictly isolate the peak corresponding to text.

  These methods work well with low computational resources but are applied mostly on gray-scale images or color channels independently. Moreover, they fail for images without any obvious peaks or with broad valleys which appear with complex backgrounds and slightly varying colors.

- **Adaptive or local binarization techniques** define several thresholds $T(i, j)$ for different image parts depending upon the local image characteristics.

  Several papers [74, 164] for video text extraction used the Niblack's method [109] where the threshold depends on local mean μ and standard deviation σ over a square window of size to define:

$$T(i, j) = \mu(i, j) + k * \sigma(i, j) \tag{3.1}$$

  where i, j, are pixel coordinates and k is an additional parameter which is set depending on the application. An extension is the method of Sauvola and Pietikäinen [130] where the threshold is defined by:

$$T(i, j) = \mu(i, j) + [1 + k * (\frac{\sigma(i, j)}{R} - 1)] \tag{3.2}$$

  where R is one more parameter to set. Sauvola and Pietikäinen suggested $k = 0.5$ and $R = 128$ for documents and a

lower threshold for stained and badly illuminated documents [129]. This adaptive technique is in use in *Mobile Reader*^{TM} [64], a mobile phone reading text from Inzisoft.

For color documents, Antani et al.[3] improved the idea of Kamel and Zhao [60] for gray-scale documents, which is to compare the average gray value and stroke width of characters. If b is the estimated width of characters, then the sliding square window over the image will be of size $2b + 1$.

For caption text, Gllavata et al. [47] created their own local thresholding based on beginning and end of text lines. They assumed fairly horizontal text lines which is not necessarily the case for NS images.

Adaptive binarizations may handle more degradations (uneven lighting, varying colors) than global ones but suffer to be too parametric which is not versatile. Moreover, these techniques still consider gray-scale images only and were mainly used for video caption text or documents with clean backgrounds.

- **Entropy-based methods**, appropriately named, use the entropy of the gray levels distribution in a scene. Li and Doermann [74] minimized the cross-entropy between the input video gray-scale frame and the output binary image. The maximization of the entropy in the thresholded image means that a maximum of information was transferred. On images including only one character, Yokobayashi and Wakahara [163] computed entropy on each channel of the CMY-converted images to select the most informative one, as the one having the largest peak of histogram. Images are very constrained with a single character and the choice of the CMY color space used for printing is not appropriate. Du et al. [27] compared Otsu's binarization and different entropy-based methods such as Pal and Pal [129]'s local entropy, joint entropy and the joint relative entropy which performs best on RGB channels independently for video caption text.

  Entropy-based techniques have been little referenced in NS context and applied only on gray-scale images or separate channels of a particular color space.

Thresholding-based methods are lightweight enough to fit low-computational resources; that is why they are preferred for par-

ticular applications with clean backgrounds for their satisfying results on gray-scale images. Nevertheless, they are not the most suitable to handle complex backgrounds, varying colors, uneven lighting and so on.

3.1.2 Grouping-based methods

The following methods group text pixels together according to certain criteria to extract text from background. Most popular techniques are clustering-based and are detailed further.

- **Region-based approaches** include spatial-domain region growing, splitting and merging, and have been extensively used in general color image segmentation with unknown content. These methods may be classified into two groups: top-down and bottom-up. The first one has been experienced in Kim et al. [66] by starting with the entire image and going towards smaller parts with differences between gray values exceeding a certain value. A merging process followed to refine results. In video captions, a bottom-up approach has been used by Lienhart and Wernicke [76]. Based on the assumption that the text contrasts well with its background, a seed around borders of text bounding box was chosen to be sure it belonged to background. With the Euclidean distance between RGB colors in a 4-neighborhood, background was extended if the distance remained below a particular value.

  In these two methods, a value was pre-defined and as all parametric methods, it is not versatile and cannot handle all degradations of NS images. Moreover region-based approaches are computationally quite expensive. However, they use spatial information which groups text pixels efficiently.

- **Learning-based approaches** have initially been designed to mimic humans by learning a training database to further recognize similar patterns. Text has interesting spatial properties and may be considered as a particular texture. Several classifiers are widely applied for pattern recognition and multi-layer perceptrons (MLP) and self-organizing

maps (SOM) are the most studied in text extraction. Neural networks, MLP or SOM, composed of linked neurons such as human brains, may model very general functions with any degree of non-linearity to separate pixels of text and non-text into two classes. In Hamza et al. [49], a cascaded approach for color historical documents with a SOM followed by an MLP was used in the training part while the trained MLP was used for testing alone. It overcame results of thresholding-based methods. Nevertheless, a training database is needed and with the wide range of NS images, this task is difficult to realize but we propose a solution in Chapter 8. Moreover it implies storage problems and labelling of the whole training database before being effective.

- **Clustering-based approaches** group color pixels into several classes assuming that colors tend to form clusters in the chosen color space. They belong to unsupervised segmentation while learning-based approaches belong to supervised segmentation. Clustering-based algorithms are the most renowned and efficient methods for NS images. They are often considered as the multidimensional extension of thresholding methods.

  The most popular method is k-means but its generalization, Gaussian Mixture Modelling (GMM), is more and more exploited. The solution in this text uses k-means clustering but attention on other partitional clustering approaches exploited in text extraction will be given in the following subsections.

  From density-based clustering to Mean-Shift: Extension of histogram-based thresholding, density-based clustering is applied on color images and needs the computation of a 3D histogram to handle color dimensions. Adjacent colors are then merged towards the nearest highest peak. The algorithm terminates when the number of desired colors is obtained. It was used on colored books and journal covers with relatively clean background and video scene text in Sobottka et al. [137] and Wong and Chen [160]. Perroud et al. [116], in one extension of Sobottka et al. [137], used a 4D-histogram with the RGB color space and

the channel of luminance Y. Histogram-based clustering is also used in the DjVu compression and display format [10]. The Mean-Shift algorithm, first created by Fukunaga in 1975 and extended by Comaniciu [19], seeks the "mode", point of highest density, of the 3D color histogram. Firstly it defines a window $\mathcal{W}$ of width r, centered randomly at a point, z_k. The mean $\widehat{Z_k} = 1/|\mathcal{W}| \sum_{j \in \mathcal{W}} z_j$ is computed and the Mean-Shift is $\widehat{Z_k} - z_k \sim \frac{f\prime(z_k)}{f(z_k)}$ where f is the density estimate. The window is then moved to $\widehat{Z_k}(\mathcal{W})$ and this loop is repeated until convergence, which is guaranteed. This successful technique has not been tested on NS text, but more generally on color segmentation. As it is publicized as the best method up-to-date, we tested it and we shall show some results in Chapter 6.

From graph theory to spectral clustering: In graph theory concept, color pixels are merged based on the minimum Euclidean distance (or another one) in a connected neighborhood to form regions in the color space. These merged pixels are represented by vertices in the graph and links between geometrically adjacent regions have weights that are proportional to the color distance between the regions they connect. They describe a hierarchy to solve by graph theory such as in [82, 125, 156]. The matrix of weights, called the affinity matrix A, is built over a sliding window $\mathcal{W}$ with $A_{ij} = \exp^{-\|x_i - x_j\|^2/2\gamma^2}$ if $i \neq j$, else equal to zero, where γ is a scaling parameter, and $x_{i....n}$ is the set of colors to cluster. It may be solved by finding a minimum of normalized cuts [108] or more generally by spectral clustering, which makes use of the spectrum of the affinity matrix. This latter method computes eigenvectors of the Laplacian matrix to have representation in the spectral space. The Laplacian matrix L is equal to $L = I - D^{-1/2}AD^{-1/2}$ where I is the identity matrix, D is the diagonal matrix whose diagonal elements are the sum of corresponding A's row, by then stacking the k eigenvectors in columns in a matrix which will be normalized, and fed to the k-means algorithm, described further in this section. k is the desired number of clusters. The main advantage of

this technique is the invariance against varying colors. The system will be further compared with this method, even if it was never tried on NS text extraction.

From k-means to GMM: k-means is considered the most used technique in clustering. The procedure follows a simple approach to classify color pixels in a defined color space through a certain number of clusters (k) fixed a priori. The main idea is to define k centroids, one for each cluster and compute a defined distance between points and centroids. Iteratively, all pixels belong to a cluster whose centroid is the nearest one. Another way to deal with clustering issues is to use a model-based approach, also called probabilistic clustering. In practice, each cluster can be mathematically represented by a parametric distribution (assumed to be Gaussian). All color pixels are therefore modelled by a finite mixture of these distributions and parameters are automatically computed with the Expectation-Maximization (EM) algorithm or one of its variants. More explanations are given in Subsection 3.1.3 and in Appendix B.

There has been little experimentation done on text extraction using other clustering methods such as fuzzy c-means, which is the extension of k-means with a degree of belonging to a cluster. As all methods can obviously not be cited in this text, the reader may refer to the survey of Berkhin [9] and only ones having a great impact on results are mentioned. In the following subsection, we focus on k-means and GMM, being the most extensively used clustering methods in text extraction.

3.1.3 Extensively used clustering methods in text extraction

- **K-means** is a simple unsupervised learning algorithm and has been extensively used in color segmentation and more specifically in text extraction. It aims at minimizing an objective function, which is the sum-of-squared error criterion J, to build representative clusters, meaning that points inside a cluster are more similar than those inside another cluster. Equation 3.3 details J with $\mathcal{M}$, the chosen metric

to compute the similarity between the n points x and the centroids c of the k clusters:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \mathcal{M}(x_i - c_j)^2 \tag{3.3}$$

The algorithm is composed of the following steps and illustrated in Figure 3.2:

1. Select k points as the initial centroids in a predefined color space
2. Select a metric $\mathcal{M}$ to assign pixels to a cluster
3. Assign each color value in the image to the cluster that has the closest centroid
4. When all colors have been assigned, recompute the positions of the k centroids
5. Repeat steps 3 & 4 until the centroids no longer move.

Different variants of k-means exist and the main ones are based on:

Choice of color space: colors to be clustered have different locations in different color spaces. Hence, k-means clustering in NS text extraction is either performed in RGB [66, 80, 147], in HSI [42, 157], in YCbCr [39] or in a dynamically decorrelated color space using principal components analysis [24].

Choice of the metric: initially, k-means computes distances between points and cluster centroids by using the traditional Euclidean distance. As far as we know, we wrote the only papers which deal with other distances in NS text extraction.

Choice of computation of centroids: as named, k-means uses the mean of points inside a cluster to compute the centroid. Other known variants such as the maximum or the medoid (leading to the k-medoid algorithm but more used for binary objects) exist and as far as we know, the mean is the only way of computing centroids, used in NS text understanding.

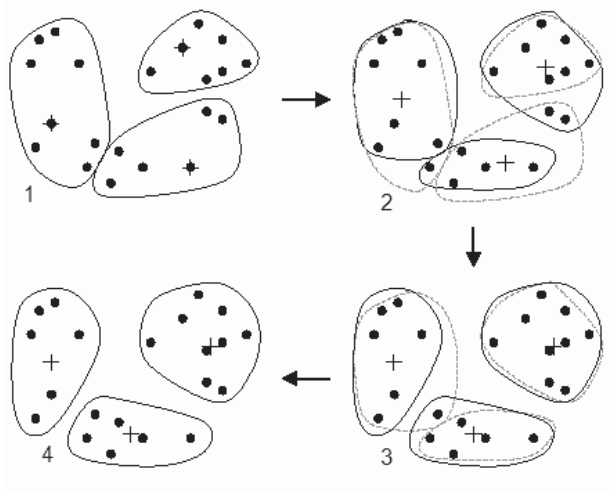


Figure 3.2: Illustration of the k-means algorithm in four points. 1: Choice of k objects and centroids and computation of clusters, 2 and 3: Computation of centroids and clusters, 4: End of algorithm when clusters no longer move.

- **Gaussian Mixture Modelling** (GMM) is also an unsupervised classifier and is used to model the probability density function of a color vector X by the weighted mixture of M basis functions (components) as:

$$P(X) = \sum_{i=1}^{M} p_i g_i(X) \tag{3.4}$$

where the weight (mixing parameter) p_i corresponds to the prior probability that color vector X was generated by each component i and satisfies $\sum_{i=1}^{M} p_i = 1$.

The basis functions g_i are chosen to be Gaussians whose probability density can be described as:

$$g_i(X|\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\Pi)^d \det(\Sigma_i)}} \exp^{(-\frac{1}{2}(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i))} \tag{3.5}$$

where μ_i and Σ_i are the mean vector and covariance matrices respectively, *det* represents the determinant of Σ and $d = 3$ for color images in a 3D color space. According to [127], Gaussian distributions are particularly well suited for NS images.

The number of clusters is pre-defined and corresponds to the number of Gaussian distributions, each representing a color distribution which is attributed to a particular object in a scene, such as text for example. To know which points belong to which Gaussians and to estimate the distribution parameter (μ_i, Σ_i), EM [21] is used as in [41, 79] for NS images and more specifically for signs, for the second reference. A detailed explanation of EM is given in Appendix B.

Similarly to k-means, several variants exist and the main ones are based on:

Choice of color space: colors represented in various color spaces lead to different segmentations and HSI is used in [161], RGB in [41], rgb in [79], and Luv in [105].

Choice of distribution parameter resolution: EM is the most used, but several extensions are also performed such as Variational EM [105] or Gibbsian EM [16].

Different ways to code the algorithm: as GMM-based clustering is computationally quite expensive, several variants of optimizations, such as [36], have been evaluated but not necessarily on text extraction.

As main drawbacks, clustering methods suffer from the need to previously set up the number of clusters and initialization variation leading to different segmentations. Problems of initialization are traditionally solved by multiple computations based on random initialization to reduce this effect towards convergent results. For the number of clusters to set, it is either pre-fixed such as in [145] or dynamically computed, with 3D histogram analysis in [66], for example.

Spatial information in clustering-based text extraction is not embedded in initial algorithms leading to non-accurate segmentations with missing pixels inside some parts of text or non-sharp edges of characters. Lopresti and Zhou [82] included local information to build the affinity matrix before using graph theory to

segment text from WWW images. For k-means, few papers proposed solutions. Fu et al. [39] included text geometry property after clustering to improve results such as in [144] with consideration on how to combine clusters to get sharp and consistent text components. Regarding GMM, spatiality is usually included by using the Potts model (also named Markov Random Field (MRF)) as GMM parameter resolution, as in [16, 159]. The main benefit is the integration of color and spatial information in the same algorithm. Nevertheless, this solution is known for its demanding resources.

3.1.4 Challenges

With this non-exhaustive literature survey on text extraction, some challenges and open task domains are raised:

- Issue relating to independence against the number of clusters
- Handling all degradations listed in Chapter 2 with a computationally interesting grouping-based method.
- Integration of spatial information to exploit interesting text properties such as alignment, similar character sizes and fonts inside a word...
- Several color spaces have been used in several papers for NS text understanding. The selection of color space is application dependent. There is apparently no color space performing better than another one for unknown NS images in a general context. Why and how can a solution group interesting properties and results of several color spaces?

3.2 Required Pre- and Post-Processing Steps for Efficient Text Understanding

Faced with multiple degradations and diversity of situations, text extraction alone is not sufficient to produce recognizable text for off-the-shelf OCR. Work on OCR itself may be done to improve results such as recognition of much degraded characters [111] without any pre-processing. Nevertheless, since the main aim is to provide a solution having satisfying performance for several kinds

of NS images, it is better to improve text quality beforehand, and only if necessary.

3.2.1 Pre-processing steps of text extraction

Low resolution, blur and complex backgrounds are the main issues relative to text extraction failures and literature is full of tried and true algorithms of resolution improvement and background removal.

- For individual still images with no a priori context, few solutions have been proposed and they were mainly based on interpolation (either bilinear or bicubic) to increase image sizes by adding interpolated pixels between existing ones which adds more information. Based on several still images of different portions of a text area, Mirmehdi et al. [99] circumvented low resolution text by mosaicing all partial images to get a higher resolution image with the whole text. Uchida et al. [150] also used mosaicing with video frames by interpreting motion speed of the camera to obtain recognizable text. To mimic behavior of presence of multiple frames in a video sequence, example-based increasing resolution are methods such as [114] which make regions of low resolution (LR) image match to higher resolution piecewise patches, present in a given database.

  For multiple frames, Wolf et al. [159] proposed to apply a bilinear interpolation on each frame of a text sequence and a higher resolution image is produced by averaging all frames. This solution has the advantage of increasing resolution and also circumventing uneven lighting effects with averaging. Nevertheless it may only work on caption text with no text motion between successive frames.

  When considering motion, **super-resolution** (SR) field is involved such as Li and Doermann [74] who assumed a pure translational model between frames for overlaid text. The motion estimation was performed using spatial-domain pair wise correlation minimizing sum of square differences between interpolated text blocks. In a driver assistance system [38], Fletcher and Zelinski used feature-based registration for the recognition of circled road signs, e.g. speed limits.

The circles were the features to be registered and normalized cross-correlation was performed on them to compute the translational motion vectors. Donaldson and Myers [23] also assumed a pure translational model and motion estimation was carried out by pairwise correlation. A Bayesian framework with a maximum a posteriori (MAP) estimator was then used for reconstruction of SR text which allowed the inclusion of a priori information to constrain errors: a bimodality prior assuming that text is bimodal and a Gibbs prior with a Huber gradient penalty function assuming that text images are locally smooth.

A pure translational model is a common assumption in most papers due to its simplicity and ease of implementation. Nevertheless, with real-scene data, it can lead to misregistration and can require a more elaborate reconstruction step. Capel and Zisserman [12] used a projective transform motion model for SR text specifically for image sequences in which the point-to-point image transformation was of enough complexity to demand such consideration. Two methods, a MAP estimator based on a Huber prior and an estimator regularized by using the Total Variation (TV) norm were proposed and compared for SR text. Only visually enhanced results were reported.

Interestingly, no affine models have been tried on text image sequences and we will suggest in this text that a simple 3-parameter affine motion model is a good representation and compromise between accuracy and overall complexity of a solution. An extended survey of SR techniques will be given in Chapter 5.

- Background removal mainly aims at reducing uneven lighting effects such as the proposition of Seeger and Dance [131] with their BST (Background Surface Thresholding) algorithm. It computed a surface of background intensities by identifying regions of LR text and interpolating background values around these regions. This was followed by an adaptive thresholding step. A similar method has been designed by Chin et al. [18] to remove ring effects of lighting in camera-based document images. The background surface, as illustrated in Figure 3.3, was obtained with a median filter over the image, followed by a histogram equalization to compute an index image where an adaptive threshold was

Figure 3.3: Illustration of the background surface thresholding algorithm. Left: original image with specular reflection, right: estimated background surface (Reproduced with kind permission from [18]).

applied depending on indexes. A wavelet-based method was proposed in Thillou and Gosselin [146]: after decomposition into several frequencies, only higher frequency information added to the lowest one for main content was kept for reconstruction. Uneven lighting is usually low frequency degradation and may be reduced with this method.

For quite clean images or camera-based documents, background removal is an efficient method which enables the use of off-the-shelf text extraction algorithms such as in [146] where the technique is followed by the global Otsu binarization. With the large diversity of NS images, we will propose a text extraction method handling simultaneously uneven lighting issues.

3.2.2 Post-processing steps of text extraction

Typical OCR fails against medium-quality extracted text having background portions, misalignment, too many adjoining characters such as text on a wavy tee-shirt where some characters are closer than others or totally connected. Hence to provide a very high quality extracted text, some post-processing is sometimes required and literature mainly counts rule-based methods and segmentation algorithms of characters into individual components.

- **Rule-based methods** are useful to remove spurious parts of non-textual extracted parts. Gatos et al. [43] defined several thresholds and global variables such as the maximum and minimum number of expected characters in a text line along with the maximum and minimum number of lines in a paragraph, while Esaki et al. [30] defined a number of rules about character sizes to remove certain parts after a global binarization method.

  Text properties, such as geometry, alignment, color and so on, differentiating text from other objects may be used to improve text extraction algorithms. Nevertheless, strict rules with thresholds are not exploitable at all for NS images.

- Classical **character segmentation** for traditional typewritten characters fails for NS images as it assumes clean conditions and particular kinds of connectedness between characters such as the projection profile method implying vertical break lines [87]. An exhaustive survey on classical character segmentation into individual components may be found in [14] and will be discussed in Chapter 7.

  With the recent emergence of NS image analysis, most papers focus on text detection and localization. When text extraction is considered, main tested images include either clean or complex backgrounds but almost without joined characters. Text on NS images such as road signs, advertisements, has to be large and easy to view with well-spaced characters. Nevertheless, more complex images may be considered with all text present in daily life such as labels on logos, brand names on clothes and so on. An example of difficult NS images with strongly joined characters is displayed in Figure 3.4. As previously mentioned, few papers proposed solutions. Among them, Karatzas and Antanacopoulos [61] worked on WWW images with difficult text and suggested a region-based method to extract text followed by a fuzzy proximity measure to add topological properties of character strokes. Chen [16] obtained more individual components by considering text extraction with spatial information by using MRF-based text extraction. Thillou and Gosselin [147] extracted text with a k-means clustering method and combined textual clusters by paying attention to pixels which connected individual components.

Figure 3.4: Sample of a word image with very connected characters on a sweatshirt.

These additional post-processing steps are dependent on quality of text extraction. Character segmentation is often required with off-the-shelf OCR to improve recognition results while rule-based methods are usually needed in addition to medium text extraction algorithm.

3.2.3 Challenges

With description of pre- and post-processing steps usually added to a text understanding algorithm, some considerations may be formulated:

- For very LR images, an increasing resolution technique is required to improve text extraction quality and consequently character recognition.
- The reduction of rule-based techniques, which are obstacles, for post-processing steps is mandatory for a versatile algorithm by definition!
- Text extraction must be of very high quality, and a good amount of work has to be completed to reduce additional steps, which are always sources of additional errors. Nevertheless, with the aim of being versatile, the steps of Section 3.2 may need to be included, but only when required, and without negatively impacting simple, clean, high resolution images or more generally well extracted text.

— CHAPTER 4 —

Text Understanding System

This chapter provides a global view of the proposed system for understanding text embedded in still images after text detection and localization. It is organized in two sections. First, an overview of the text understanding system is briefly introduced and the second section mainly presents and discusses the characteristics of the databases used in this text.

4.1 Text Understanding Chain

When several consecutive steps are useful for a given task, first works are based on a sequential scheme. It is also true of the text understanding system with all steps described in Chapter 3. Nevertheless some papers merge some steps such as Kusachi et al. [71] which recognized Kanji characters directly from the original image by using a training database with numerous forms of each character with large deformations and partial parts of characters. The image was divided into several squares which are then recognized as a part of Kanji characters or not. Negishi et al. [106] extracted and recognized an isolated or a connected character directly from a scene to circumvent false negative detection of connected characters. They extracted edge or curve features and used voting, similar to the generalized Hough transform to simultaneously realize the two steps. The main drawback is the consideration of one character only in each process. In a particular application for robot navigation in a known environment, room numbers were also recognized from the whole image with template matching in

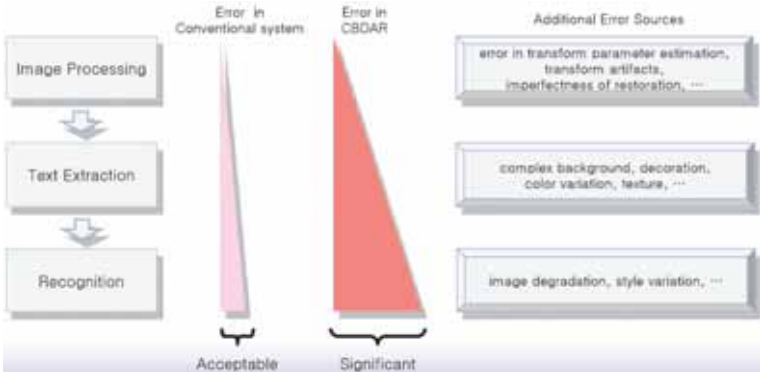


Figure 4.1: From [65], comparison of accumulation of errors along with a sequential text understanding chain between conventional systems and camera-based systems (CBDAR stands for camera-based document analysis and recognition).

Iwatsuka et al. [56]. Impressive results were presented in Tu et al. [149] where text was simultaneously detected, extracted and recognized by combining bottom-up learning-based algorithms and top-down generative models using the Data Driven Markov Chain Monte Carlo algorithm to make models fit to pixels.

These algorithms are computationally expensive or driven by a particular application only. Part of our motivation is to build an efficient text understanding system with lightweight algorithms to fit within mobile devices' resources (such as PDAs) as they will be intensive future users of these systems.

To circumvent degradations of NS images acquired by cameras of different qualities, we opt for a feedback-based system. In-Jung Kim, in [65], described the impact of accumulation of errors in camera-based systems compared to conventional scanning acquisition as explained in Figure 4.1. Hence, it highlights the necessity to choose a feedback-based system to dynamically correct errors and as soon as possible.

Almost all steps of the proposed system are reinforced by information of the following step to refine results as described in Figure 4.2 and numbered in the similar way.

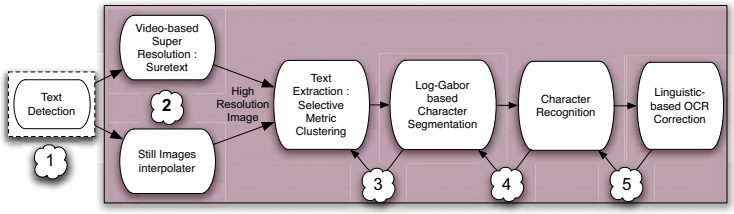


Figure 4.2: Proposed text understanding system: parts in pink will be detailed in following chapters.

1. Due to computational efficiency, text detection and localization must be performed as soon as possible. In many papers such as [143], text detection is processed on LR images to decrease computation time, hence the increasing resolution algorithm can be done afterwards. Note that text detection and localization are out of scope of this text.

2. The second main step increases resolution of LR images or LR video frames. The SURETEXT algorithm detailed in Chapter 5 assumes a simple affine model for motion registration and uses Teager filtering to enhance high frequency information before reconstructing the higher resolution image.

3. The text extraction step using selective metric clustering takes advantage of the subsequent step of character segmentation. This latter step uses spatial information to segment textual components into individual characters and to refine text extraction results.

4. To properly segment text into individual characters and to dynamically choose parameters of log-Gabor filters dedicated to this task, the following recognition step is used to validate character segmentation.

5. Lastly, the final feedback is performed between recognition and correction. By adding linguistic information to the best OCR outputs, correction of recognition errors is carried out to increase results.

4.2 Material and Databases

Several databases are used to test the system or to produce results in order to validate or invalidate a method. The use of several databases enables us to highlight versatility and robustness of proposed solution and to emphasize independence of algorithms from a particular set of images.

DB-ICDAR [1]: This database was initially created for a competition of word recognizer in the scientific community. Almost no results were provided compared to the ICDAR 2003 text locating competition. Hence recognition algorithms were obviously not mature enough to be presented in a contest. This database of 2266 images has the great advantage to be public for (future) comparisons with other papers and has been built with several cameras with different resolutions, unknown a priori. Images are all in color and compressed in JPEG format.

DB-Sypole: Research for this text was partially used in a national project named Sypole aiming at building a mobile text recognizer for the visually impaired. This database was mostly taken by blind people and we keep only images with text. For most people, it was their first time taking a picture, and some pictures of hand or walls, for example, do not contain text. We completed this database to increase the number of pictures to 500. The single-sensor CMOS cameras used were of various brands: HP Photosmart Mobile Camera (HP), Pretec Compact Camera or Pocket Loox Fujitsu Siemens F20 (Loox), and 1.3 Megapixel cameras embedded on top of a personal digital assistant or already integrated in it. Images were saved in either BMP or JPEG formats. This database includes images with English and French text. 300 images with French text were used to test the correction method.

DB-WWW: Web images have some similar issues as NS images such as low resolution and complex backgrounds and we extended tests of the algorithm on LR text of 150 images. 100 images include French text as well to test the final step of correction.

DB-VideosPDA: Even if this text focuses on still images, we recorded 16 short gray-scale videos of text with slight mo-

Figure 4.3: Samples of the various databases used in this text. From top to bottom, left to right: samples from DB-ICDAR, DB-WWW, DB-VideosPDA, DB-Sypole.

tion between frames, actually equal to hand shaking when holding a camera-enabled PDA to test the super-resolution method and to add information to still images. Video sequence duration is about $5-7$ seconds each. Videos were taken with either the QVGA HP (288×352) or CIF Loox cameras (288×352) with a frame rate of 5 frames per second which is low and representative of low quality cameras.

Samples of each database are illustrated in Figure 4.3. All databases include color NS images except DB-WWW where, instead, artificial text is embedded. They are populated with images taken with different cameras of different resolutions in different situations with clean and complex backgrounds of daily life. Moreover, neither of the databases have simulated data or constrained lighting to conform to reality as close as possible and to give representative results! Some images may be considered as "easy" with even lighting, clean background, no blur and homogeneous character sizes and fonts, whereas others are "very difficult" with complex backgrounds, specular highlights, artistic fonts and so on, which enables us to highlight the versatility of the system. Gray-scale versions of databases are considered in Chapters 5 and 7 to exploit only intensity values y.

The transformation is assumed to be a linear combination of the red, green and blue intensity values, R, G, and B and given by:

$$y = 0.299R + 0.587G + 0.114B \tag{4.1}$$

Figure 4.4: Differences between a manual text locating system (in red) and an automatic one (in white): some characters may be cut such as the word "SALE" and false detections may appear (bus doors)(Reproduced with kind permission of Springer Science and Business Media from [84]).

Detection of text areas were either already performed in DB-ICDAR or manually done in DB-VideosPDA. For DB-Sypole and DB-WWW, detection was both manually and automatically performed using A.Chen's algorithm [83], available on Internet. By providing test images, users receive evaluation of A. Chen's method through a dedicated website in an XML format. We used this automatic text locator because it was announced as one of the best entries of the ICDAR 2005 text locating competition and it is also publicly available for comparison with other works. Moreover, automatic methods are not as perfect as manual ones as displayed in Figure 4.4 and the system aims at being robust against errors in text detection and inaccuracy of the text areas' bounding boxes.

For easier comparisons with other papers, in some evaluation part, a commercial OCR is mentioned. Hence, each time an off-the-shelf OCR is used in tests, it is referred to ABBYY FineReader 8.0 Professional Edition Try&Buy [1]. Similarly, when Matlab is mentioned, it is related to Matlab version 7 R14 [54].

[1] http://france.abbyy.com/download/?param=46440

— CHAPTER 5 —

Resolution Enhancement

The quest for high resolution images or image sequences from a cheap and small acquisition system is a challenge rooted deeply in both hardware and software. While hardware advances in leaps and bounds in terms of more powerful yet smaller footprint processors, sensors, and memory, the progress of software and appropriate algorithms requires longer-term research and development. Due to the increased use of embedded low-resolution imaging devices, such as handheld PDAs and mobile phones, coupled with the need to extract information accurately and quickly, resolution enhancement techniques are quickly becoming a crucial step in the field of text recognition.

If the Nyquist criterion is assumed to be respected when an object image is sampled by a camera array, it is theoretically possible to perfectly reconstruct the original light field by an interpolation function. Nevertheless, in practice, in the presence of noise or more obviously in single-sensor cameras, the Nyquist criterion is not satisfied leading to subsampling which causes artifacts. Hence, the aim is to produce as perfect a high resolution (HR) image as possible. Section 5.1 will investigate results for still images while Sections 5.2, 5.3 and 5.4 will be dedicated to algorithms in *super-resolution* (SR) field - that is using several frames to reconstruct the HR image - , the proposed SURETEXT algorithm and finally evaluation and results.

5.1 Resolution Enhancement for Still Images

Resolution enhancement of one still image is challenging by definition as there are no other sources of information. An example could be to increase resolution of text on an advertisement poster, acquired by a low-priced camera. Conventional interpolation algorithms, such as nearest neighbor, bilinear (with a linear kernel) or bicubic (with a cubic kernel), can be classified by basis functions. Those algorithms have been developed by assuming that there is no correlation among adjacent pixels in the imaging sensor, no motion blur, and no aliasing in the process of sub-sampling. Since these assumptions are not true in general low-resolution (LR) imaging systems, conventional interpolation algorithms are not appropriate.

More complicated algorithms such as vector median filtering [86] which handles multidimensional vectors, like representation of colors, or the Bimodal-Smoothness-Average [26] which takes advantage of text image properties, give slightly better results. Another category called *SISO* (Single-Input Single-Output) super-resolution enables to recover HR information missing in a single LR image by training models to learn piecewise correspondences between LR and possible HR information to form a SR image. Nevertheless, improvement is not really significant for text extraction algorithms compared to the additional computation time of these methods. To circumvent missing high frequency information, we chose a method similar to [120], a conventional interpolation by a factor of 2, the bicubic method, followed by an edge sharpener, as described in Equation 5.1.

$$I_{improved} = I + \lambda I" \tag{5.1}$$

To enhance high frequency information, we use unsharp masking which adds to the original I a portion (λ) of its second derivative $I"$ computed using a Laplacian filter. A larger factor of interpolation may lead to degradations, due to aliasing or compression artifacts.

Nevertheless, resolution enhancement of still images cannot achieve the same improvement as when using multiple frames. Figure 5.1 describes the effects of this solution: on right, the hole inside the "A" of "JAIN" is recovered and the result is crispier leading to better text extraction and recognition. However, as shown

Figure 5.1: Difference between non-enhanced and enhanced extracted text. From left to right: original image and its extracted text, resolution-enhanced image and its extracted text: the hole inside 'A' has been recovered after resolution enhancement. Text extraction is processed with the SMC algorithm described in Chapter 6.

Figure 5.2: Impact of resolution enhancement on very LR still images: original image (top-left), its extracted text (top-right), resolution-enhanced image (bottom-left) and its extracted text (bottom-right). Text extraction is processed with the SMC algorithm described in Chapter 6.

in Figure 5.2, very LR images lead to unsatisfying results. Due to aliasing effects, unsharp masking may highlight this kind of noise along with character edges. Hence, the size of the Laplacian operator is dependent on character size. For more versatility, it is better to increase resolution for small upscaling only. For highly LR images, dedicated methods may be designed. Nevertheless, if information from multiple frames is present, results will be drastically improved. For more details on enhancement of still images, the reader may refer to the recent survey of Van Ouwerkerk [151].

5.2 Super-Resolution for Video Frames[1]

As stated in the previous section, the combination of multiple frames, which are sources of additional information, may effi-

[1]Subsections 5.2.1 and 5.2.2 are under the form of a book chapter that we have written in [95] where the reader may find finer details.

ciently enhance resolution. This time, a mobile phone camera may be used to capture one or more lines of the advertisement poster with several frames obtained manually (inducing shaking) for only a few seconds. The result is also a LR image sequence. This could be possibly sent to a server for transformation into ASCII text or be done on the fly on the phone if (one day) it is enabled to do so.

5.2.1 Context of super-resolution algorithms

Most SR algorithms deal with the integration of multiple LR frames to estimate a higher resolution image. The most common term of reference for multiple frame super-resolution found in the literature is *Multiple-Input Single-Output (MISO)* or *static super-resolution*. A recent focus of SR research relates to *dynamic super-resolution* which is aimed at reconstructing a high quality set of images from low quality frames, often referred to as *Multiple-Input Multiple-Output (MIMO) super-resolution*. This approach is also known as *video-to-video super-resolution*. For example, applications can be found in video enhancement captured by surveillance cameras to increase the general visibility and acuity of a recorded event; but *MIMO* SR is out of scope of this text. For more details on general super-resolution and its applications, the reader is referred to [114].

In this chapter the focus is on the application area of text analysis: how can SR be used in the generation of higher quality text images that can be more accurately interpreted by in-house or off-the-shelf OCR software?

NS text suffers from different degradations and by using multiple frames of a video sequence and static SR techniques, most of these degradations can be minimized or even suppressed. For example, in character recognition, text fonts are assumed to have sufficient resolution to be reliably recognized by OCR. For document images, 300 dpi is plenty for satisfactory recognition and that means characters can occupy an area as large as 40×40 pixels. However, in video frames, a resolution of 320×240 is very common and therefore text may well be rendered no larger than 10×10 pixels, hence the enhancement of spatial resolution becomes important[1].

[1] These numbers are based on the assumption that the acquisition device is at a sensible, realistic distance from the text.

The SR problem is usually modelled as the reversal of a degradation process. This is an example of an inverse problem where the source information (SR image) is estimated from the observed data (LR images). Solving an inverse problem generally requires first constructing a forward model. Most imaging devices can be described as a camera lens and aperture which produce blurred images of the scene contaminated by additional noise from various sources: quantization errors, sensor measurement or model errors. For an SR image x of size $M \times N$ and a set of K LR images y_k, the observation model can then be expressed as:

$$y_k = DB_kW_kx + n_k \tag{5.2}$$

where W_k is a $M \times N$ warp matrix which maps the HR image coordinates to the LR coordinates and represents the motion that occurs during image acquisition, B_k is a $M \times N$ blur matrix caused by the optical system, the relative motion during the acquisition period and the point spread function (PSF) of the LR sensor, D is the decimation matrix of size $(M \times N)^2/(L \times P)$ where L and P are the subsampling factors in the horizontal and vertical directions respectively, and finally n_k is the associated noise. Usually D and y_k are known and are inputs in the SR algorithm. Using columnwise reordering and by stacking the frame equations, Equation 5.2 can be rewritten as:

$$y = Hx + n \tag{5.3}$$

where H represents all the degradations, i.e. $H = DB_kW_k$ for all k. Super-resolution is a computationally intensive problem which involves several thousand unknowns. For example, super-resolving a sequence of just 50×50 pixel LR frames into a 200×200 SR image by a factor of 4 in each direction involves 40000 unknown pixels. As mentioned, SR is an inverse problem and is ill-conditioned due to the obvious lack of LR frames and the additional noise. Hence matrix H is under-determined and regularization techniques may have to be used to overcome this problem in the image super-resolution process.

Super-resolution algorithms require several processing stages, from motion estimation through reconstruction to deblurring, possibly involving regularization along the way. An overview is shown in Figure 5.3. These stages can be implemented consecutively or simultaneously depending on the reconstruction methods chosen (we will come across examples of these later).

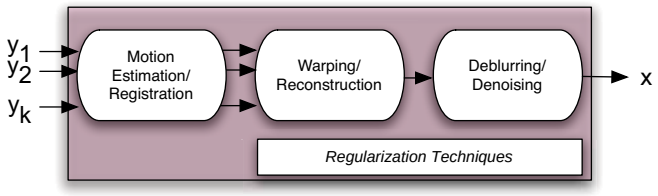


Figure 5.3: General scheme for super-resolution.

Motion estimation/registration An important key to successful super-resolution is the existence of change between frames, e.g. by motion in the scene or through ego motion. For example for scene motion, consider a fixed camera video surveillance scenario monitoring cars for licence plate recognition; low resolution and low quality image sequences arising due to weather conditions and changing illumination can be enhanced to increase the chance of character recognition. In the context, an example of camera motion would be a handheld camera-enabled PDA capturing a text document for a short period. The difference between the frames arising through hand jitter would result in a suitable set of frames for super-resolution.

Motion estimation is then the first step in SR techniques and motion parameters are found through some form of registration, i.e. the relative translations, rotations and other transformations that define an accurate point-to-point correspondence between images in the input sequence. Usually, each frame is registered to a reference one (most commonly the first) to be able to warp all frames into a single higher resolution image in the reconstruction stage. An alternative would be to register each frame against its preceding frame but consecutive temporal errors can accrue leading to inaccurate results.

An error in motion estimation induces a direct degradation of the resulting SR image. The artifacts caused by a misaligned image are visually much more disturbing to the human eye than the blurring effect from interpolation! Nevertheless, we will see in Section 5.3 how to deal with a limited number of motion estimation outliers. Clearly, the perfor-

mance of motion estimation techniques is highly dependent on the complexity of the actual motion and the model used to represent it.

The two parameter translational model is often enough to reasonably represent scene motion in many different applications, not least one where a handheld device is used for a short period to capture some text. Indeed according to [32], the model approximates well the motion contained in image sequences where the scene is still and the camera is moving. Moreover, for sufficiently high frame rates most motion models can, at least locally, be approximated by this simple and low cost model. However, the assumption of a pure translational model is not always valid and can result in significantly degraded performance. A regularization technique or a deblurring process must then be applied to constrain or correct motion estimation errors (or a higher order motion model employed).

Correlation is the main path to a solution in the translational model and both frequency and spatial domain based variations have been applied in text-related applications. The main advantages in using correlation in the frequency domain are fast computation and illumination-invariance in phase space.

Phase correlation is a well-known method in frequency domain analysis and was applied by [164] for text SR. The main steps in phase correlation are based on the shifting property of the Fourier transform. Hence, if the motion vector is assumed to be only the translation $(\Delta x, \Delta y)$ between two frames, then

$$f_{t+1}(x, y) \approx f_t(x - \Delta x, y - \Delta y) \tag{5.4}$$

for frames at times t and $t + 1$. After applying the Fourier transform:

$$F_{t+1}(u, v) \approx F_t(u, v) \exp^{-2\pi j(u\Delta x + v\Delta y)} \tag{5.5}$$

Then the cross-power spectrum CPS of F_t and F_{t+1} can be defined as:

$$CPS = \frac{F_t(u, v)F^*_{t+1}(u, v)}{|F_t(u, v)F^*_{t+1}(u, v)|} \approx \exp^{-2\pi j(u\Delta x + v\Delta y)} \tag{5.6}$$

where F^*_{t+1} is the complex conjugate of F_{t+1}. The maximum of the Fourier inverse of CPS is then at $(\Delta x, \Delta y)$.

In the spatial domain, Donaldson and Myers [23] used pairwise correlation over the whole image with quadratic interpolation and a least-squares fit to determine the translation vector for each observed LR frame. Li and Doermann [74] performed sub-pixel registration by first bilinearly interpolating frames and then by using correlation minimizing Sum of Square Difference (SSD) between text blocks.

The affine motion model assumes planar surfaces and an orthographic projection. It is clearly more involved than the pure translational model and requires the computation of a warp matrix accounting for rotation, scale and shear as well as a translational vector term. Interestingly no solicitation of this model can be found in application to text SR within a *MISO* framework. This is rather surprising given that text capture at a close distance, where images in a sequence would mostly differ by translation and rotation, is an ideal scenario for applying the affine motion model. Li and Doermann [74] in fact mentioned that the general 6-parameter affine model should be used in their text analysis application, but resort to a pure translational model due to the difficulty in obtaining a sufficient set of corresponding points to compute the affine parameters. They applied the translational model to multiple frames to enhance overlaid movie credits that move up the screen or ticker text that moves across the screen.

For rigid scenes, the 8-parameter projective model provides the most precise parameters to account for all possible camera motions. Capel [13] applied this model for text SR. He first computed interest point features to sub-pixel accuracy using the Harris corner detection algorithm [50]. Then using RANSAC [37] to deal with outliers, a Maximum Likelihood estimator was used to compute the homography matrix between successive frames. Shimizu et al. [134] computed motion estimates between each frame pair by assuming that the consecutive frames exhibit only small pure translational motion differences. To reconstruct all the frames into a SR image, motion estimation parameters have to be estimated against a single reference LR image. Hence, simultaneous 8-parameter projective estimation using an 8D hyperplane

and parabola fitting was then performed to refine the initial motion parameter estimates.

Optical flow is another motion estimation approach not yet applied to text super-resolution. No doubt researchers in the field will turn their attention to it soon especially as increasing computational power will be able to deal with such an intensive technique, particularly for more complex motion models.

In summary, there are few SR techniques in which motion estimation is dealt with in depth and most works concentrate on reconstruction and regularization. If necessary, motion registration parameters are assumed to be known or integrated as errors from an additive Gaussian noise process.

At the extreme end, highly complex, non-rigid, non-planar motions are very difficult to investigate and can occur in text analysis; an example is for text that appears on a curled page or on a moving person's loose t-shirt. Such examples need very special treatment and are beyond the scope of this chapter.

Warping and Reconstruction The stage after motion estimation comprises of some way of bringing together all the input LR images into a coordinate frame that reconstructs a SR output. There are several methods that divide the reconstruction process into "grid mapping and interpolation" or "interpolation and fusion". There are also other methods that simultaneously reconstruct and deblur.

Grid Mapping and Interpolation - This is the most intuitive reconstruction process involving mapping onto a higher resolution grid followed by bilinear or higher order interpolation; first motion estimation parameters are applied to map LR pixel values into the SR sampling grid. This is shown in the left of Figure 5.4 with three LR frames where the second frame is a translation of the first and the third frame is a translated and rotated version of the first. For pure translational motion, this algorithm is often called *'Shift-and-Add'*. Nevertheless, some pixels are unknown or missing because of a lack of LR frames and have to be interpolated to build and refine the reconstruction. The advantage of grid mapping and interpolation is in its low computational cost making real-time applications possible. On the other hand, only

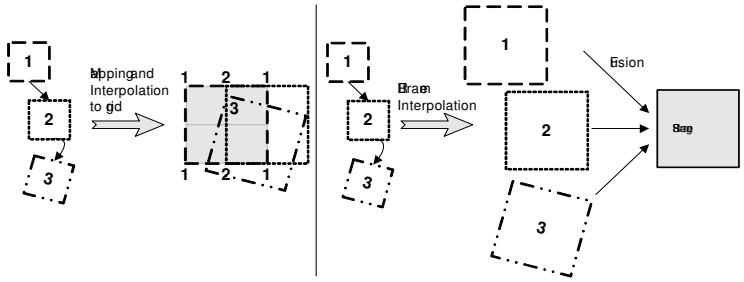


Figure 5.4: Schema of two reconstruction techniques. Left: grid mapping and interpolation, right: interpolation and fusion.

the same blur and noise for all LR frames can be assumed, which reduces the overall performance.

Interpolation and Fusion - Warping, using the motion estimation parameters, is applied between each independent LR frame and the first one, instead of mapping to a SR grid as in the previous scheme. Then, linear or non-linear interpolation methods are performed to increase the resolution of each LR frame separately. Finally, a fusion between all the resolved frames results in a SR image at the resolution of the interpolated LR frames. This is shown in the right of Figure 5.4. Depending on the fusion method, not all frames contribute to reconstruct pixels in the SR image. In the particular example of median fusion, only one of the LR frames is used for each reconstructed pixel. Hence, motion estimation outliers, salt and pepper noise, etc. are discarded in the reconstruction process. Farsiu et al. [32] recommended the median for this purpose. For text enhancement in digital video, Li and Doermann [74] used bilinear interpolation followed by averaging of the interpolated frames. In order to be invariant against illumination changes, Chiang and Boult [17] fused only the edges of each warped text frame into a reference interpolated image with a median filter. Interpolation and fusion is fast and robust to outliers but it can result in the appearance of some artificial effects in the super-resolved image due to the nature of the fusion process.

Frequency-domain reconstruction - This particular form of reconstruction is very often the continuation of frequency-domain motion estimation in the case of pure translational model assumption. It was first derived by Tsai and Huang [148] and was the first implemented SR reconstruction method, also called *alias-removal* reconstruction. Assuming that LR images are under-sampled, the translations between them allows an upsampled SR image to be built based on the shifting property of the Fourier transform and the aliasing relationship between the continuous Fourier transform of an original SR image and the discrete Fourier transform of observed LR images [114]. Frequency-domain reconstruction has never been implemented in a SR text application. The major advantage is its simplicity but only global translational models can be considered.

Iterative Back-Projection (IBP) - IBP reconstruction was first introduced by Irani and Peleg [55] and has found much use in mainstream SR reconstruction. Given knowledge of the imaging process (PSF model and blur parameters amongst others) relating the scene to the observed image sequence, it becomes possible to simulate the output of the imaging system with the estimate of the original scene. The simulated images may then be compared with the observed data and a residual difference error found. Next the process is repeated iteratively to minimize this error. Thus this technique comprises two steps: simulation of the observed images and back-projection of the error using an adequate kernel to correct the estimate of the original scene. Several works, such as [12], run comparisons against this method using text images, however they are for demonstration only and are not specifically designed for text. IBP methods have no unique solution due to the ill-posed nature of the inverse problem. In fact, minimizing the error does not necessarily imply a reasonable solution and a convergent iteration does not necessarily converge to a unique solution.

Projection Onto Convex Sets (POCS) - The POCS method describes an alternative iterative approach but with more flexibility to include prior knowledge about the solution into

the reconstruction process. Convex constraint sets have first to be defined to delimit the feasible solution space for SR restoration containing all LR images. Constraints can be various but have to represent data in the best way to yield desirable characteristics of the solution. For example, one constraint could be to enable only a range of pixel values. Other more complex constraints can be defined depending on the objectives and the application. The solution space of the SR restoration problem is the intersection of all the constraint sets. This method was initially proposed by Stark and Oskoui [138] and then extended by Patti et al. [115].

POCS can be considered as a generalization of the IBP method and has never been investigated in SR text. It has several disadvantages such as the non-uniqueness of the solution, slow convergence and high computational cost, but provides the flexibility to enable the inclusion of a priori information.

Maximum A Posteriori estimator (MAP) - The MAP approach provides a flexible and convenient way to model a priori knowledge to constrain the solution. Usually, Bayesian methods are used when the probability density function (pdf) of the original image can be established. Given the K LR frames y_k, and using the Bayes theorem, the MAP estimator of the SR image x maximizes the a posteriori pdf $P(x|y_k)$, i.e.:

$$x_{MAP} = \arg\max_x P(x|y_k) = \arg\max_x \frac{P(y_k|x)P(x)}{P(y_k)} \quad (5.7)$$

The maximum is independent of y_k and only the numerator needs to be considered.

MAP reconstruction in SR text has been seen in depth investigation. Capel and Zisserman [12] used an image gradient penalty defined by the Huber function as a prior model. This encourages local smoothness while preserving any step edge sharpness. Donaldson and Myers [23] used the same Huber gradient penalty function with an additional prior probability distribution based on the bimodal characteristic of text. The MAP estimator with the Huber penalty prior term provides slightly smoother results. Robustness and flexibility in degradation model estimation and a priori knowledge of

the solution are the main benefits of the MAP estimator approach to the ill-posed SR problem. On the other hand, the main disadvantages are the high computational costs and the complexity of implementation.

Assuming that the noise process is Gaussian white noise and a convex prior model, MAP estimation ensures the uniqueness of the solution. Elad and Feuer [28] proposed a general hybrid SR image reconstruction which combines the advantages of MAP and POCS. Hence, all a priori knowledge is put together and this ensures a single optimal solution (unlike the POCS only approach).

Regularization Regularization techniques can either be used during the reconstruction process or the deblurring and denoising step as shown in Figure 5.3.

Super-resolution image reconstruction is an ill-posed problem because of a recurrent lack of LR images and ill-determined blur operators. To stabilize the problem and find a relevant solution, it is necessary to incorporate further information about the desired solution and this is the main purpose of regularization. Using Equation 5.3, a regularization cost function $\Lambda(x)$ can be added such that:

$$\sum_{k=1}^{K} \|y_k - Hx\| + \lambda\Lambda(x) \tag{5.8}$$

where λ is the regularization parameter for balancing the first term against the regularization term. The choice of x is then obtained by minimizing Equation 5.8.

An optimal regularization parameter must be chosen carefully and there are various methods for its selection. Tikhonov regularization (Λ_T) and Total Variation (TV) regularization (Λ_{TV}) are popular techniques for this purpose expressed respectively as $\Lambda_T(x) = \|\Gamma x\|_2^2$ where Γ is usually a high pass operator and $\|.\|_2$ is the L_2 norm and $\Lambda_{TV}(x) = \|\nabla x\|_1$ where ∇ is the gradient operator and $\|.\|_1$ is the L_1 norm. Tikhonov regularization is based on the assumption of smooth and continuous image regions while TV is not and preserves the edge information in the reconstructed image. Hence, TV is recently becoming the more preferred regularization method for denoising and deblurring to reach a stabilized solution in SR reconstruction.

Regularization methods are very complementary to the MAP estimator as the cost function can be seen as a priori information. Capel and Zisserman [12] implemented both of the cost functions in their MAP reconstruction process. Farsiu et al. [33] compared various reconstruction techniques, among which were grid mapping and cubic spline interpolation, Tikhonov regularization, and bilateral TV regularization (extension of TV regularization).

To obtain acceptable results in complex images, a regularization technique is often required during the reconstruction process but not all reconstruction methods can include spatial a priori information, e.g. frequency domain reconstruction methods.

The second main use of regularization techniques is for denoising and deblurring and can be applied on still images as well. The process is the same: for a blurred and noisy image, a regularization technique can be performed to recover the original data from the degraded one as an inverse process. Moreover, if the high pass operator Γ in the Tikhonov cost function is the identity matrix, then the method is the well known inverse Wiener filtering.

Deblurring and Denoising Causes of blur are the optical system, relative motion during the acquisition stage, and the PSF of the sensor as well as from interpolation and registration errors. Noise can come from salt and pepper noise in the LR images as well as from misregistration outliers. SR algorithms generally include an independent post-processing step to deblur and denoise the final image. Usually, standard deconvolution algorithms, such as Wiener deblurring or blind deconvolution, are applied. Nevertheless, if the PSF is unknown and the LR images are strongly motion-blurred, a robust estimation of the PSF and the direction of the motion blur must first be performed before applying deblurring methods.

If the blur estimation is accurate enough, efficient deblurring can occur simultaneously during reconstruction. Recovering an image with an estimated PSF is a mathematically ill-posed problem; that is why regularization techniques described previously are used to solve it. However, knowledge of the blurring process is the best route to the cure and

blur identification is sometimes included in the reconstruction procedure and refined iteratively. Chan and Wong [15] proposed blind deconvolution based on TV regularization by iteration. In another example, Chiang and Boult [17] performed local blur estimation by modelling a blurred edge with a step edge and a Gaussian blur kernel. During the reconstruction process, the unknown standard deviation of the kernel was estimated iteratively with the edges extracted previously. Hence, edge pixels were re-estimated using the edge model. The purpose was then to fuse the edge information into a reference interpolated image to overcome illumination sensitivity.

Denoising can be approached via classical post-processing routes, for example after all LR frames are warped and interpolated separately, image fusion can be applied at each pixel position across the available frames. Additionally, noise removal can be implemented, e.g. Zhao et al. [166] used a trimmed mean while Farsiu et al. [32] applied a median filter.

5.2.2 Color super-resolution text

Color remains a ripe area for investigation in general SR, let alone for the text SR application. The most common solutions apply monochrome SR algorithms to each of the color channels independently or simply the luminance channel only, such as [55]. An interesting work in the text SR area is that of Shimizu et al. [134] who proposed a reconstruction step which took into account color information by demosaicing. After motion estimation from non-demosaiced LR frames, extended IBP reconstruction was used, reinforced by the evaluation of the difference between the simulated LR frames and the original LR frames. Hence, Bayer sampling was used instead of classical downsampling. In the SURETEXT algorithm detailed next section, only gray-scale LR sequences are considered due to the high computational cost of color-based algorithms. Nevertheless, for more accurate results and with the advance in hardware, such considerations will disappear soon.

5.3 SURETEXT - Super-Resolution Text

As mentioned earlier in this chapter, recent advances in hardware and sensor technologies have led to handheld camera-enabled devices such as PDA or smartphones giving rise to new potential applications, such as handy text OCR. In this section we present an experimental approach to reconstructing a higher resolution image, from the low resolution frames obtained from a PDA, by applying a novel super-resolution technique with the aim of getting a better response from standard off-the-shelf OCR software. Hence, the primary goal is not to nicely super-resolve text but to enhance it in order to get higher recognition rates.

The database used for experiments is DB-VideosPDA, introduced in Chapter 4. No a priori knowledge of parameters such as camera sensor noise, PSF and so on was used. Hence, the approach is independent of camera models.

The method described here enhances the classical SR approach by complementing it with high frequency information extracted from the LR frames using an unsharp masking filter called the Teager filter. The classical SR approach can be said to consist of the stages shown in the upper row in Figure 5.5. The lower row shows the added Teager filtering process. Motion parameters are estimated for the LR frames using Taylor decomposition, followed by a simple RANSAC-based step to discard obvious outlier frames. The frames are then warped onto a high resolution grid and bilinearly interpolated to obtain a preliminary SR result. The original frames (except the outliers) are then put through the Teager filter to generate a high pass set of frames which are also warped and interpolated for a secondary SR result. The two resulting SR images are then fused and median denoising is applied to smooth artifacts due to the reconstruction process to obtain the final SR image. We shall call this method SURETEXT (SUper-Resolution Enhanced TEXT) and the entire process is outlined next.

5.3.1 Motion estimation using the Taylor series

For motion estimation we apply Taylor series decomposition as presented in [62] who used it to register frames to correct atmospheric blur in images obtained by satellite. This approach fits very well to text capture with a quivering hand since a shaking hand can produce slight random motions and the approximation

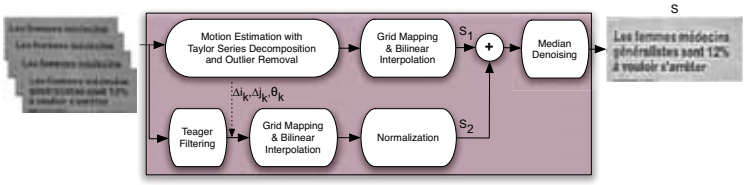


Figure 5.5: Schema of SURETEXT.

computed by Taylor series decomposition can be suitable due to the small motion amplitudes involved. Initially a pure translational model was used but this led to too many (small) misregistration errors to adequately and reasonably correct afterwards. A significant improvement was noticed when stepping up to a 3-parameter affine motion model (Δi_k, Δj_k, for horizontal and vertical translation, and θ_k for rotation). Given K frames with $k = 1, .., K$, the motion between a frame y_k and the first frame y_1 can be written as:

$$y_k(i,j) = y_1(i\cos\theta_k - j\sin\theta_k + \Delta i_k, j\cos\theta_k + i\sin\theta_k + \Delta j_k) \quad (5.9)$$

Replacing the sin and cos terms respectively by their 1^{st} and 2^{nd}-order Taylor series expansion:

$$y_k(i,j) \approx y_1(i + \Delta i_k - j\theta_k - i\frac{\theta_k^2}{2}, j + \Delta j_k + i\theta_k - j\frac{\theta_k^2}{2}) \quad (5.10)$$

This can be approximated using its own 1^{st}-order Taylor series expansion:

$$\begin{aligned} y_k(i,j) \quad \approx \quad & y_1(i,j) + (\Delta i_k - j\theta_k - i\frac{\theta_k^2}{2})\frac{\partial y_1}{\partial i} \\ & + (\Delta j_k + i\theta_k - j\frac{\theta_k^2}{2})\frac{\partial y_1}{\partial j} \end{aligned} \quad (5.11)$$

The optimum motion parameter set $\mathbf{m}_k = (\Delta i_k, \Delta j_k, \theta_k)$ can then be estimated by solving this least-squares problem:

$$\begin{aligned} \underset{\Delta i_k, \Delta j_k, \theta_k}{\arg\min} \quad & \textstyle\sum_{i,j}[y_1(i,j) + (\Delta i_k - j\theta_k - i\frac{\theta_k^2}{2})\frac{\partial y_1}{\partial i} + \\ & (\Delta j_k + i\theta_k - j\frac{\theta_k^2}{2})\frac{\partial y_1}{\partial j} - y_k(i,j)]^2 \end{aligned} \quad (5.12)$$

To find $\mathbf{m}_k$, the minimum can be computed by obtaining the derivative with respect to Δi_k, Δj_k and θ_k and setting it to zero. Neglecting the non-linear terms and the small coefficients, then the following 3×3 system must be resolved:

$$\begin{pmatrix} A & B & C \\ B & D & E \\ C & E & F \end{pmatrix} \begin{pmatrix} \Delta i_k \\ \Delta j_k \\ \theta_k \end{pmatrix} = \begin{pmatrix} \sum (y_k(i,j) - y_1(i,j)) \frac{\partial y_1}{\partial i} \\ \sum (y_k(i,j) - y_1(i,j)) \frac{\partial y_1}{\partial j} \\ \sum (y_k(i,j) - y_1(i,j))(i \frac{\partial y_1}{\partial j} - j \frac{\partial y_1}{\partial i}) \end{pmatrix} \tag{5.13}$$

with $A = \sum \frac{\partial y_1}{\partial i}^2$, $B = \sum \frac{\partial y_1}{\partial i} \frac{\partial y_1}{\partial j}$, $C = \sum (i \frac{\partial y_1}{\partial j} - j \frac{\partial y_1}{\partial i}) \frac{\partial y_1}{\partial i}$, $D = \sum \frac{\partial y_1}{\partial j}^2$, $E = \sum (i \frac{\partial y_1}{\partial j} - j \frac{\partial y_1}{\partial i}) \frac{\partial y_1}{\partial j}$, $F = \sum (i \frac{\partial y_1}{\partial j} - j \frac{\partial y_1}{\partial i})^2$. After the motion estimation stage in SURETEXT, outlier frames corresponding to incorrect motion estimates are removed (see subsection 5.3.3). This allows the warping and bilinear interpolation (by a factor of 4) of the remaining N LR images to obtain an initial SR image S_1 as:

$$S_1 = \mathcal{I}(\sum_{k=1}^{N} W_{\mathbf{m}_k} y_k) \tag{5.14}$$

where $W_{\mathbf{m}_k}$ is the warp matrix for each LR frame y_k using motion estimation parameter set $\mathbf{m}_k$, and $\mathcal{I}$ is the interpolation function.

5.3.2 Unsharp masking using the Teager filter

SURETEXT attempts to recover the high frequencies in the LR images such that the relevant high frequencies such as character/background borders can be highlighted but impulsive perturbations can not. Non-linear quadratic unsharp masking filters can satisfy these requirements. For example, the 2D Teager filter which is a class of quadratic Volterra filters [100] can be used to perform mean-weighted high pass filtering with relatively few operations. Using the set of N corresponding original frames, Teager filtering is performed to obtain $y_k^\tau, (k = 1, ..., N)$ as the set of filtered images. For example, for any image y:

$$\begin{aligned} y^\tau(i,j) = 3y^2(i,j) &- \frac{1}{2} y(i+1,j+1) y(i-1,j-1) \\ &- \frac{1}{2} y(i+1,j-1) y(i-1,j+1) \\ &- y(i+1,j) y(i-1,j) - y(i,j+1) y(i,j-1) \end{aligned} \tag{5.15}$$

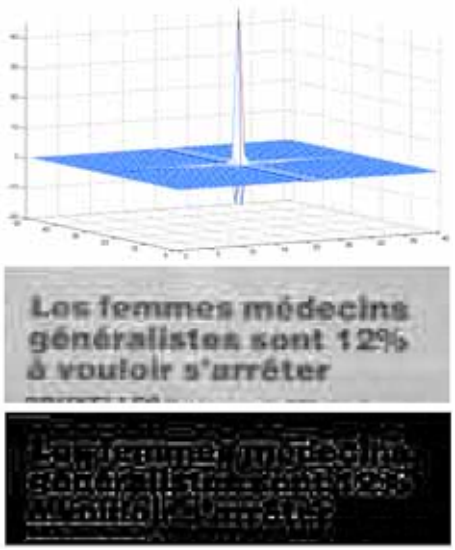


Figure 5.6: Visualization of the 2D Teager filter (left) and results on an image (right) with on top the initial LR image and on bottom the Teager-filtered output.

This filter enables us to highlight character edges and suppress noise. The shape of the Teager filter is shown in Figure 5.6 along with an example image and its Teager filtered output. Next, the frames can be warped using the same corresponding motion parameters $\mathbf{m}_k$ to reconstruct a secondary SR image S_τ:

$$S_\tau = \mathcal{I}(\sum_{k=1}^{N} W_{\mathbf{m}_k} y_k^\tau) \tag{5.16}$$

This is then normalized to provide:

$$S_2(i,j) = \frac{S_\tau(i,j) - min(S_\tau)}{max(S_\tau) - min(S_\tau)} \tag{5.17}$$

Also see the lower row in Figure 5.5. The final SR output image S is then:

$$S = med(S_1 + S_2) \tag{5.18}$$

where *med* is median denoising applied after fusion of the motion corrected representation with the motion corrected high frequency content.

5.3.3 Outlier frame removal

A few algorithms deal with outlier frame removal, especially those including MAP-based reconstruction or regularization techniques, as these methods aim at reducing the presence of outliers. However, it is argued in [6] that for large magnification factors, regularization suppresses useful high-frequency information and ultimately leads to smooth results. In our method, errors occur during motion estimation between frames if a text line is incorrectly registered with a neighboring one. A frame corresponding to incorrectly estimated parameters in $\mathbf{m}_k$ should therefore be dropped from further analysis. In this set of experiments, it was found that Δi_k or θ_k rarely caused any errors, whereas misregistrations frequently occurred on the vertical translations Δj_k leading to results such as that shown in Figure 5.7. The left example in Figure 5.8 shows a plot of Δj_k points in which an outlier value can be rejected after linear regression. However, there may be consecutive sets of outlier frames, hence outliers can be detected by fitting a RANSAC-based least squares solution to the *differences* between vertical translations (illustrated on the right of Figure 5.8). Outlier frame rejection not only reduces the number of frames processed, but most importantly removes the need to apply regularization techniques during or after the reconstruction process. Note, this can easily be performed in SURETEXT on all parameters in $\mathbf{m}_k$.

5.3.4 Median denoising

In Figure 5.9 a zoomed view of a text document is presented to emphasize the importance and effect of (a) Teager filtering and (b) the median denoising stages. The second image shows a pure interpolation of the original frame. The third shows the interpolation result of all the frames in the sequence and hence is the result of $med(S_1)$ only. The fourth image is the result of $(S_1 + S_2)$ illustrating significant improvement when the Teager processing pipeline shown in Figure 5.5 is employed. Median denoising becomes necessary as the reconstruction result $(S_1 + S_2)$ alone is not smooth enough with errors arising from all the earlier stages of motion registration, warping, and interpolation. The resulting artifacts are objectionable to the human eye and would affect OCR. A 3×3 neighborhood median filter was applied in all text images in this work. The last image in Figure 5.9 shows the final result obtained from Equation 5.18.

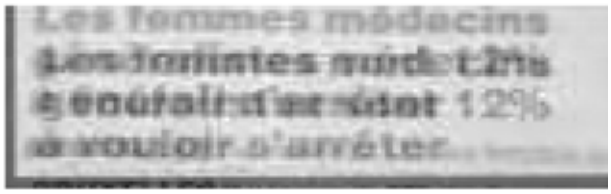


Figure 5.7: Fusion of two misregistered frames.

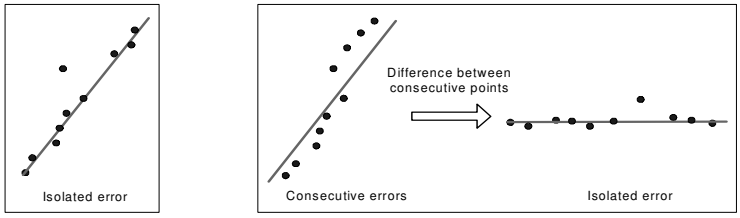


Figure 5.8: Left: an isolated Δj_k error, right: consecutive Δj_k errors result in wrong estimation, hence Δj_k *differences* must be examined.

5.4 Experiments and Results

5.4.1 Evaluation of SURETEXT

The impact of Teager filtering can be further emphasized as follows. The top-left image in Figure 5.10 shows the results of a classical *MISO* approach (the same as just the top row of the diagram in Figure 5.5, i.e. $med(S_1)$ only). In comparison, the top-right image shows Teager filtering of a set of LR frames fused together and then combined with an interpolated original frame, similar to the edge enhancement concept suggested in Chiang and Boult [17]. The bottom image shows the result of SURETEXT which exhibits more sharpness and readability.

Figures 5.11 and 5.12 present more text images with and without the Teager stage to highlight the usefulness of this filter. In the zoomed examples in Figure 5.12, while OCR of all the SR images will recognize the characters in both methods, however note the difference in quality after Otsu binarization where the SURETEXT produces a much sharper and better defined set of

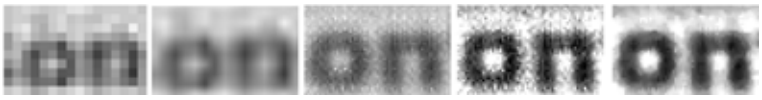


Figure 5.9: Results of each step of SURETEXT. From left to right: original LR frame, bilinear interpolation applied on one LR frame, SR output without using Teager-filtered frames (S_1), SURETEXT without the denoising stage ($S_1 + S_2$) and the complete SURETEXT method.

Figure 5.10: Results highlighting the importance of order of each step of SURETEXT. First row: left: classical approach ($med(S_1)$), right: Teager-filtered frames after median fusion with an interpolated original frame. Second row: the result from SURETEXT.

characters with Teager filtering than without. The Teager filter is very good as a quadratic, unsharp masking filter. Other similar filters such as the rational filter of Ramponi [126] may also be capable of achieving similar results.

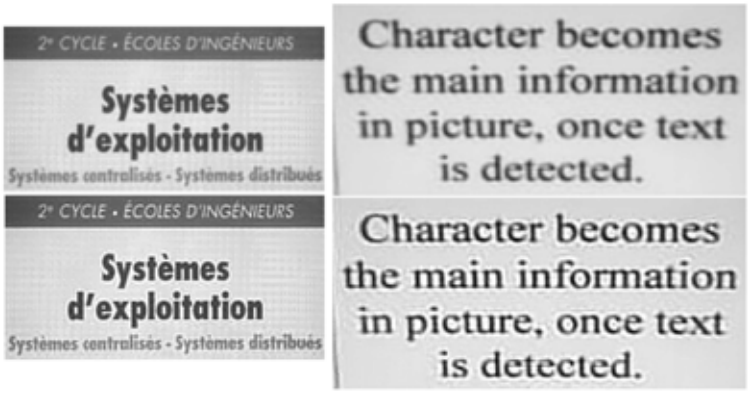


Figure 5.11: SR using the classical approach on top and the proposed method on bottom.

Figure 5.12: Two zoomed in SR results comparing the classical approach (on left of each result) and the proposed method (on right of each result) and their binarized images (bottom row).

Moreover, Figure 5.13 specifies the importance of enhancing high frequencies before reconstruction. Comparison is done with the SURETEXT framework and a classical approach in general super-resolution with unsharp masking applied after denoising. Unsharp masking code of Matlab was used for this comparison. Results are crispier when edges are enhanced before reconstruction than after reconstruction, concluding that reconstruction is performed more easily on sharp edges.

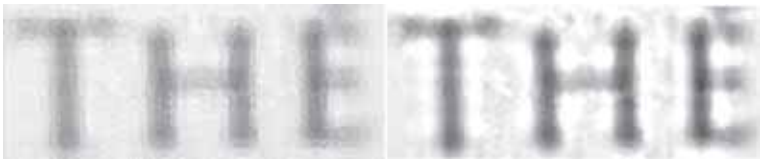


Figure 5.13: Importance of Teager filtering before reconstruction. Left: unsharp masking applied after a classical approach without Teager filtering, right: SURETEXT result.

For all results, previously mentioned in this section, a number of 15-20 frames were used to compute the enhancement. A smaller number may be considered but obviously with lower performance, while a larger number may lead to motion estimation errors. It is important to note that the motion estimation is based on the first frame and larger motion may be observed after a long time. In that case, a pyramidal decomposition is needed to use this SURETEXT algorithm.

Finally, percentage recognition rates based on several natural scene text video sequences are shown in Table 5.1 for comparison of the classical approach (C), a framework the same as SURETEXT but with a standard Laplacian unsharp masking filter (L) and SURETEXT, as proposed here, with the Teager filter (S). The results demonstrate much better performance by SURETEXT at 86.5% accuracy on average, computed on the number of correctly recognized characters, showing unsharp masking to be clearly an important additional step to generating an SR image while also being less sensitive to noise than a standard unsharp masking filter such as the Laplacian. Examples where SURETEXT results are lower than with the Laplacian masking is mainly due to the commercial OCR, used in experiments. Results looked similar but for any reasons, recognition results were slightly different. On these cases, an home-made OCR similarly performs.

As illustrated in [94], SURETEXT may be performed on other sets on video sequences and not necessarily with text. Effectively, some tests have been done on faces presenting enhanced results. Nevertheless, as SURETEXT is based on emphasizing high frequency information and especially edges, results are more appropriate for data with strong edges required such as text. On faces, some edges may appear on cheeks, which may be not expected after improving resolution.

Table 5.1: Comparative OCR accuracy rates (%). Indexes of methods C, L and S represent the name of each sequence.

Sequence	C_{1-8}	C_{9-16}	L_{1-8}	L_{9-16}	S_{1-8}	S_{9-16}
1/9	48.1	72.7	78.8	81.8	78.8	90.9
2/10	75.2	72.5	94.3	88.8	92.9	93.8
3/11	65.2	84.4	56.5	93.8	78.3	93.9
4/12	77.7	13.0	84.4	56.5	86.0	60.9
5/13	95.1	81.5	100.0	88.8	100.0	81.5
6/14	66.6	90.5	83.3	100.0	91.6	100.0
7/15	75.0	45.5	79.4	54.5	86.4	90.9
8/16	79.3	63.2	79.3	78.9	79.3	78.9
Avg.	69.1		81.2		86.5	

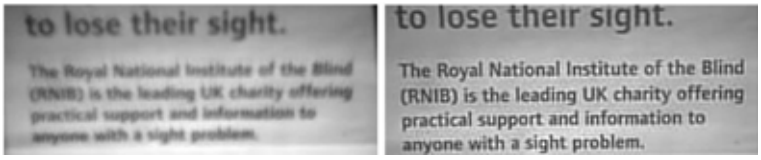


Figure 5.14: Left: SR image obtained with the algorithm of Li and Doermann [74], right: our method.

5.4.2 Comparison with state-of-the-art SR algorithms

In Figure 5.14 the result of SURETEXT is compared to the method in Li and Doermann [74] in which a simple translational model was used for text enhancement. Bearing in mind that Li and Doermann's method was developed for text primarily moving in vertical and horizontal directions, nevertheless this comparison shows that the use of an affine model is minimally necessary in the type of applications referred to in this chapter. The registration errors in the left image of Figure 5.14 make it very difficult for interpretation by OCR analysis.

Farsiu et al. [32, 33] developed a Matlab software, named MDSP [98] standing for Multi-Dimensional Signal Processing, with several algorithms and their variants to compare results with in-house methods. Figure 5.15 details visual performances between the four best MDSP methods for SR text and SURETEXT. Among the list of algorithms, for comparison, we chose the *'Shift-*

Figure 5.15: Comparison of SURETEXT and four best algorithms of the MDSP toolbox [98] for SR text. From left to right: *'Shift-And-Add'* method (SA), Bilateral SA method, Median Gradient method with L_2 regularization, IBP with L_1 regularization and the SURETEXT result.

and-Add' method described in Subsection 5.2.1, the same method with the additional step of outliers removal using bilateral frame rejection, IBP method with a median operator and L_2 norm regularization, and IBP method with L_1 norm regularization. Motion estimation is identical for all algorithms and uses a pyramid registration on gradient images. Details of algorithms may be found in [98]. To be independent of our data set, we compared results with a video sequence provided in the software. All parameters were set to default to get an automatic method. Figure 5.15 shows that SURETEXT presents more contrasted characters with less noise and artifacts.

5.4.3 Computation cost

SR algorithms are known to be quite computationally expensive and SURETEXT is not really an exception. Its computation time is much lower than very sophisticated methods including Bayesian framework and regularization techniques and to compare with the MDSP software developed also in Matlab, computation time of SURETEXT is similar, even slightly lower.

It has no sense to give computation time references in Matlab as code is not optimized at all especially for an embedded purpose. Nevertheless, 78% of SURETEXT is spent during the reconstruction procedure as detailed in Table 5.2. Note that median denoising is run using the appropriate function (*medfilt2*) of Matlab.

Incremental motion estimation may be performed to decrease computation time of motion estimation. Motion parameters are not initialized for each frame and benefits from motion estimation

Table 5.2: Occupation time for each step of SURETEXT(%).

Steps	Occupation time
Motion Estimation/Registration	17%
Teager Filtering	1.5%
Outlier Frame Removal	3%
Reconstruction	78%
Median Denoising	0.5%

of the previous frame, assuming that motion dynamics is smaller than frame rate. Following that, motion parameters may be updated for each frame. About reconstruction, a detailed study is required to interpolate missing pixels efficiently. A pyramidal structure in several levels to reconstruct HR images may be useful. For example an enhancement factor of 4 may be divided into two factors of 2. Code optimization for this part is extremely needed.

5.5 Conclusions

Resolution enhancement for still images represents a challenging issue compared to results obtained with multiple frame integration:

- Interpolation or *SISO* methods, except for those based on offline patches learning from a set of frames, lead to resolution-increased images but need an edge sharpening technique to circumvent smooth outputs of conventional interpolation methods.
- These basic functions are inherently lightweight and fast to perform, hence preferred for this reason.

The SURETEXT method in Section 5.3 is typical of a general approach to SR text in which frame sequences must at first be adequately registered and subsequently enhanced to increase the rate of character recognition:

- A simplified affine model is preferred and compared with methods which assume a pure translational model, the results are improved.
- A Teager-based filtering is used as an edge sharpener to enhance results and makes reconstruction steps more efficient.

- An outlier frame removal step is added in order to make basic reconstruction methods efficient without regularization techniques and to also be independent of camera sensors.
- A median denoising enables the smoothing of artifacts coming from different sources and also alleviates uneven lighting or poor outdoor conditions effects.
- Detailed experiments and results enable the conclusion of the efficiency of our method. Each step has been carefully validated, leading to OCR rates increase and positive comparisons with recent state-of-the-art SR algorithms.
- Methods such as SURETEXT must not be computationally expensive in order to fit into PDA and mobile-phone devices; however, such limitations are expected to be overcome as advances in hardware and software continue to surpass expectations. Nevertheless, for the short-term, computation optimizations are required and still need to be sent to a remote server.
- Sophisticated methods touched upon in Section 5.2 may lead to better results, but such results are obtained at the cost of assuming the camera's point-spread-functions or circumventing approximate assumptions with more steps and a higher computation time.
- SURETEXT is not completely dedicated to text enhancement but instead, to enhancing objects-with-edges. More consideration on text may be added with spatial information for example. Nevertheless, the difficult compromise between fast SR techniques and high-level processing on text must be noted.

— CHAPTER 6 —

Text Extraction

In this chapter, we aim at giving solutions to issues stated in Chapter 2 and discussed briefly in Section 2.3: how to handle varying colors? how to efficiently combine luminance and chromaticity properties for text extraction and how to be independent of color spaces while simultaneously using magnitude and orientations of colors?

6.1 Impact of Color Spaces and Clustering Algorithms

6.1.1 Is there a better color space for NS text extraction?

Based on the impractical and non versatile definition of color spaces, the main goal is to circumvent effects of a predefined observer at either 2° or 10° and to assess the behavior of several color spaces in text extraction.

Until recently, only transformations from RGB to YUV or YCbCr were computationally interesting. More conversions can now be considered with efficient algorithms and powerful hardware. Nevertheless, one keeps in mind that a compromise has to be done on text extraction quality and conversion resources required, meaning that if a result is just perceptibly better but very heavy to compute, the choice will be towards the fastest solution with almost similar results.

The RGB color space has a Riemannian nature, meaning that it is not a uniform space and perceived differences among colors cannot be assessed directly from a classical Euclidean distance between colors. As digital camera sensors displayed images in RGB, the color space will be one of the tested arenas. For its invariance to uneven illumination, the normalized rgb space will also be considered.

The XYZ color space defined specifically for standardization is not perceptual as well, not even realistic considering image formation such as described in Section 2.2. Hence, we will not use this color representation.

On the contrary, Lab and Lch are two perceptual color spaces defined in different ways, respectively by Cartesian and polar coordinates. These two color spaces will be tested in the context of NS images in this text. Moreover, Lch already gave satisfying results in text extraction in Mancas-Thillou and Gosselin [88]. Luv, which is often compared to Lab for its similar results, will also be considered, since as Wesolkowski [158] stated that it is invariant to viewing and illumination directions and surface orientation.

Ruderman et al. [127] have shown that for natural image ensembles, the resulting axes have simple forms and interpretation, forming a new color space, introduced in Chapter 2 as $I_1I_2I_3$. It will also be interesting to compute results with this color space.

The last type of color spaces using hue, invariant to certain highlights and shadows, such as HSV, is a controversial category of color spaces. It is apparently very efficient for NS images as main problems are about specular surfaces and interreflections, and has already been used in text extraction by Garcia and Apostolidis [42]. Nevertheless, Poynton [122] stated, among others, that this type does not match the same lightness perception as Lab for example, and introduces visible discontinuities in color space due to different computations around 60° segments of the hue circle. The behavior of HSV on a large NS data set will become apparent in this section.

Combinations of color spaces by reducing or increasing the 3D color representation is sometimes performed to take advantage of several color spaces as in [2, 153]. To be invariant against illumination changes, HS, ab, uv, ch color spaces from respectively, HSV, Lab, Luv and Lch will be considered after removing the lightness component, along with RGBHS, RGBab, RGBuv, RGBch, abch, and uvch to include perceptual meaning inside RGB or to combine Cartesian and polar coordinates.

Other exotic color spaces such as SCT, a spherical color transform [121], have also been tested but results are not relevant enough to be mentioned. Results presented in Subsection 6.1.3 attempt to show there is no better color space for NS images and RGB is a sufficient color representation for realistic text extraction, as tested on a large data set.

6.1.2 Considerations on different clustering algorithms

To assess the behavior of all chosen color spaces, several clustering methods, such as k-means, GMM, Mean-Shift, and spectral clustering, will be tested based on either their massive use in text extraction or their promising properties.

1. K-means has the main advantage to be easy to implement and fast to compute with the triangle inequality [29] for a chosen metric equal to the Euclidean distance, for instance.

   One major drawback is to fix the number of clusters to build. As mentioned by Berkhin [9], there is no way to find a good number of clusters, with respect to a particular application. In this case, the logical number of clusters is 2 for text and non-text color values. Nevertheless, with the large variety of NS images, it is better to define 3 clusters, one for background, one for textual foreground and another one for noise, which may be useful to handle complex backgrounds with varying colors or text of different colors. For "clean" documents with monochrome text on a uniform background, the third cluster represents text edges, always slightly different from main text due to image formation, subsampling effects around the edges and so on.

   The number of clusters may be dynamically declared such as in [66] or iteratively using Message Minimum Length (MML) or Bayesian Information Criterion (BIC) measures as explained, among others, in the large survey of Berkhin [9]. In this case, as text is already detected and text is an object that has to be easily readable for humans, three clusters lead to very satisfactory results. Some previous tests have been done using the elbow criterion, which is an experimental measure to choose the right number of clusters for a given data set. It is achieved by plotting the percentage of variance explained by the clusters against the number of clus-

ters, where the first clusters add more information than the subsequent ones, and drawing a specific angle in the graph, corresponding to the best number of clusters. Results on a large portion of DB-ICDAR show that the best number of clusters was three. This conclusion is only true for already detected text areas.

For non-detected text areas, the number has to be chosen dynamically. Nevertheless, the algorithm of Figueiredo and Jain (FJ) [36], briefly summed up by the following lines, may be used as it computes a variant of MML to find the correct number of clusters.

2. GMM, often considered as a generalization of k-means, is a technique solved by the likelihood of the data given the clusters and EM enables to find a local maximum of the likelihood within a similar iterative procedure:

   (a) Calculate cluster probability for each instance (expectation step)

   (b) Estimate distribution parameters based on the cluster probabilities (maximization step)

   The detail of EM algorithm may be found in Appendix B.

   For evaluation, we will use the FJ algorithm [36], which overcomes several major weaknesses of the EM algorithm: firstly, the number of Gaussians is dynamically obtained during the estimation process, where components becoming singular are annihilated and secondly, it starts with a large number of components to tackle the initialization issue of the EM algorithm. Components are iteratively reduced by the bottom, meaning that components which are less probable and "closer" to each other are merged.

3. Extensively used in color segmentation, the Mean-Shift algorithm finds the point of highest density in a 3D color histogram. Several parameters need to be defined such as the window size (here, 20 pixels: this value was chosen experimentally on the data set as the best one) to compute Mean-Shift and the minimum area to consider (here, 20 pixels).

4. Spectral clustering has emerged recently as a popular clustering method that uses eigenvectors of the affinity matrix

derived from the data. It has one major advantage over k-means, which is robustness against clusters not corresponding to convex regions. K-means is known to create elliptical clusters and it is similar to GMM, which assumes that the density of each cluster is Gaussian. In Subsection 6.1.3 we evaluate the algorithm of Ng et al. [108], briefly introduced in Chapter 3. To compute the affinity matrix, a neighborhood of 40 pixels is considered with the Euclidean distance to build 3 clusters among the numerous steps of the algorithm. The scaling parameter, which controls how rapidly the affinity matrix falls off with the distance between two colors, is set at 0.04. These parameters have been set experimentally on our data set as the best ones.

For each clustering algorithm, parameters are fixed to equally compare all encountered algorithms based on versatility.

6.1.3 Evaluation of color representation with state-of-the-art clustering algorithms

In this subsection, results of text extraction using several color spaces and clustering algorithms are stated to highlight the properties and impact of each color space and clustering method behavior. For each table of results, DB-ICDAR, DB-WWW, and DB-Sypole are considered to appreciate the different results if any. Color spaces were either developed with Matlab or already built-in in the software. About k-means, a particular code in C language was developed for computation time explanation.

To assess the usefulness of a color representation, Precision and Recall are defined enabling the evaluation of the text extraction quality:

$$\text{Precision} = \frac{\text{Correctly extracted characters}}{\text{Total extracted characters}} \tag{6.1}$$

$$\text{Recall} = \frac{\text{Correctly extracted characters}}{\text{Total number of characters}} \tag{6.2}$$

Taking the (weighted) harmonic average of Precision and Recall leads to the F-score [152]:

$$F = (1 + \beta^2)\frac{\text{Precision} \times \text{Recall}}{\text{Recall} + \beta^2\text{Precision}} \tag{6.3}$$

where β is a weight to balance the importance of either Precision or Recall.

Precision measures the quality of extraction while Recall measures the quantity of high quality extraction. "Correctly extracted characters" means characters which are extracted without noise or missing parts of the character. The F-score is often chosen for an understanding purpose. It is actually easier to compare single values than a couple of values to know which ones are best. We choose $\beta = 1$ to give an equal weight to Precision and Recall. In NS text extraction, both have an important impact as the aim is to properly extract a large number of characters. As no ground truth is available, visual inspection is performed and results are given in Table 6.1.

An ROC ("Receiver Operating Characteristics") curve, plotting true positive rates (equal to Recall) against the false positive rates (equal to the ratio between the number of incorrectly extracted characters and the total number of non-characters) may also be used for evaluation. ROC curves and Precision and Recall computations have been proven to be intimately linked [20]. Nevertheless, the total number of non-characters is difficult to assess as the the whole background has to be considered as non-characters and false positive rates have less meaning in text extraction than Precision and Recall. If several components are incorrectly labeled as text, they could be discarded in the following steps of character recognition and correction. Moreover, as evaluations and text extraction are based on colors and are applied on constrained textual areas, this case is quite rare.

Better results are obviously obtained with natural scenes with more colors, such as the ones in DB-ICDAR and DB-WWW. Several characters of DB-WWW were not correctly extracted due to very low resolution of some images. It is important to note that recognition may compensate those approximate extractions. About color spaces, the RGB color space with k-means performs better in terms of F-score. RGB has already been proven its global efficiency in several papers for different applications using large data sets [88, 135]. Nevertheless, some interesting results are obtained with the hybrid color spaces, RGBch and RGBHS. Hue information is integrated either with polar coordinates (ch)

Table 6.1: Precision, Recall and F-score measures for several color spaces in a k-means clustering framework.

	DB-ICDAR			DB-WWW		
	P	R	F	P	R	F
RGB	0.90	0.88	**0.89**	0.81	0.78	**0.79**
rgb	0.66	0.57	**0.61**	0.60	0.52	**0.56**
Lab	0.70	0.22	**0.33**	0.67	0.20	**0.31**
Luv	0.61	0.24	**0.34**	0.60	0.24	**0.34**
Lch	0.85	0.26	**0.40**	0.80	0.24	**0.37**
$I_1I_2I_3$	0.62	0.54	**0.58**	0.60	0.52	**0.56**
HSV	0.66	0.57	**0.61**	0.60	0.52	**0.56**
HS	0.37	0.13	**0.19**	0.40	0.14	**0.21**
ab	0.81	0.39	**0.53**	0.82	0.39	**0.53**
uv	0.64	0.56	**0.60**	0.62	0.54	**0.58**
ch	0.87	0.84	**0.85**	0.82	0.79	**0.80**
RGBHS	0.74	0.64	**0.69**	0.68	0.59	**0.63**
RGBab	0.82	0.79	**0.80**	0.72	0.69	**0.70**
RGBuv	0.31	0.13	**0.18**	0.36	0.15	**0.21**
RGBch	0.72	0.62	**0.67**	0.70	0.60	**0.65**
abch	0.58	0.42	**0.49**	0.55	0.40	**0.46**
uvch	0.65	0.49	**0.56**	0.60	0.45	**0.51**

	DB-Sypole			**Average**
	P	R	F	F
RGB	0.91	0.89	**0.90**	**0.86**
rgb	0.58	0.50	**0.54**	**0.57**
Lab	0.68	0.20	**0.31**	**0.32**
Luv	0.56	0.22	**0.32**	**0.33**
Lch	0.84	0.26	**0.40**	**0.39**
$I_1I_2I_3$	0.58	0.50	**0.54**	**0.56**
HSV	0.62	0.54	**0.58**	**0.58**
HS	0.35	0.12	**0.18**	**0.19**
ab	0.77	0.36	**0.49**	**0.52**
uv	0.60	0.52	**0.56**	**0.58**
ch	0.82	0.79	**0.80**	**0.82**
RGBHS	0.67	0.58	**0.62**	**0.65**
RGBab	0.74	0.71	**0.72**	**0.74**
RGBuv	0.30	0.13	**0.18**	**0.19**
RGBch	0.71	0.61	**0.66**	**0.66**
abch	0.51	0.37	**0.43**	**0.46**
uvch	0.60	0.45	**0.51**	**0.53**

Figure 6.1: Some improved results with inclusion of hue information. From left to right: original image, result inside RGB, result inside RGBHS. The three clusters are displayed with black, white and gray colors.

Figure 6.2: Segmentation conflict inside the RGBch color space. From left to right: original image, result inside RGB and result inside RGBch.

or with the H information. Figure 6.1 shows examples of color clustering which fail inside the RGB color space and work better inside the RGBHS one. Similar results are obtained with the RGBch color space. Meanwhile, some conflicts between pixel values and difficulty to properly cluster in the case of divergent information may be observed in the RGBch color space for other DB-ICDAR images in Figure 6.2.

For other clustering algorithms, detailed results are not relevant enough, moreover several issues such as noise and non versatility prevents them to be used.

Figure 6.3 shows the behavior of Mean-Shift with two different images and two different window size values, meaning that this latter value has to be set dynamically according to each image. This is the same conclusion for spectral clustering with neighborhood size must vary depending on characters size to extract.

GMM presents too noisy results (Figure 6.4). This may be reduced with post-processing techniques. This method may also be coupled, as stated in Chapter 3, with spatial information using Potts model, to circumvent noisy results as well. In addition, GMM and Potts model are computationally expensive compared

Figure 6.3: Impact of window size in the Mean-Shift algorithm. From left to right, first row: original image, result with window size of 50 and 20, second row: original image, result with window size of 5 and 20.

Figure 6.4: Noisy results (on right) for GMM-based clustering.

to k-means. Nevertheless, GMM may be a very efficient tool for text detection, and can be refined afterwards with more accurate techniques such as what is proposed in Section 6.3.

6.2 Role of Metrics in K-means

In a previous evaluation of color spaces (Subsection 6.1.3), RGB yielded satisfying results in terms of F-score but still presented problems in terms of handling varying colors such as shown in Figure 6.5, which are better handled by perceptual color spaces or hue-based ones as illustrated in the same figure, shown on the right. In order to handle more NS images in RGB, we investigate the role of metrics in the k-means clustering algorithm.

6.2.1 Definition of some metrics, either distances or similarities

Several metrics, either distances or similarities, have been designed to be used in k-means in different fields requiring unsupervised classification, such as the Minkowski metric, generalization

Figure 6.5: Example of failures of text extraction with RGB compared to HSV. From left to right: original image with varying colors and uneven lighting, extraction result inside RGB with k-means and extraction result inside HSV with k-means.

of the traditional Euclidean distance, the Canberra distance or the normalized correlation for example. Several other measures exist and the reader is referred to [119].

To understand which ones to test and use, it is first more relevant to define differences between distance and similarity. Distance gives results in the range $[0; \infty[$ with 0 indicating no difference between colors while similarity is in the range of $[0; 1]$ with 1 indicating that colors are identical. Nevertheless, one may note that distances may be converted in similarities such as the Euclidean distance D_{eucl} leading to the "Euclidean similarity" S_{eucl}:

$$D_{eucl}(x, y) = \sqrt{\sum_{i=1}^{3}(x_i - y_i)^2} \text{ and } S_{eucl}(x, y) = \exp^{-D_{eucl}(x,y)^2/2} \tag{6.4}$$

with x and y representing two different 3D color vectors.

To include hue information inside the RGB color space and to fill one of the issues stated in Chapter 2 to take advantage of color orientation as suggested in MacAdam ellipses, angle-based similarities will be considered as:

$$\text{Hue} = \begin{cases} 2\pi - \theta & \text{if } B > G \\ \theta & \text{otherwise} \end{cases} \tag{6.5}$$

$$\theta = \arccos\left(\frac{1}{2}\frac{(R-G)+(R-B)}{((R-G)^2+(R-B)(G-B))^{1/2}}\right) \tag{6.6}$$

where hue is represented by an angle ranging in value from 0 to 2π.

The angle-based similarity has been previously used for edge detection or color segmentation by Wesolkowski [158] by exploiting the sine of the angle instead of cosine as the dynamic range for the latter's small angles is small compared to the former's, for color classification by Hild [52], and for vector directional filtering by Lukac et al. [86], for example. The k-means using the Cosine similarity (named S_1 in the following lines) is also called spherical k-means and has been extensively used in another field related to text, called text clustering where each document is represented as a vector of word occurrences [140]. Two documents with the same proportions of occurrences but different lengths are often considered identical and may be grouped in the same cluster using the Cosine similarity.

In a very large survey, Hild [52] detailed properties on several color similarity measures and five of them, those having particular expected behaviors for text extraction, will be considered, among other similarities. Note that θ shall represent the angle between two colors vectors x and y:

- S_1 also called "Cosine similarity", "Vector Angle" or "Normalized correlation" is the most popular angle-based similarity and is defined by:

$$S_1(x,y) = \frac{x.y}{\|x\|.\|y\|} = \cos(\theta) \tag{6.7}$$

  S_1 is sometimes defined by $1-\cos(\theta)$ but to match the given similarity definition, we preferred this description.

- S_2, which was not tested by Hild, is the suggestion of Wesolkowski [158]:

$$S_2(x,y) = \left(1 - \left(\frac{x.y}{\|x\|.\|y\|}\right)^2\right)^{1/2} = \sin(\theta) \tag{6.8}$$

- S_3 is defined by:

$$S_3(x,y) = \frac{\|x\|\cos(\theta) + \|y\|\cos(\theta)}{(\|x\|^2 + \|y\|^2 + 2\|x\|\|y\|\cos(\theta))^{1/2}} \tag{6.9}$$

- S_4 is defined by:

$$S_4(x,y) = \frac{\cos(\theta)((\|x\|^2 + \|y\|^2 + 2\|x\|\|y\|\cos(\theta))^{1/2}}{\|x\| + \|y\|} \tag{6.10}$$

S_1, S_3 and S_4 have been chosen for their insensitivity to brightness changes in the image.

- S_5, which has been tested in [91] for its compact support, is defined by:

$$S_5(x,y) = \cos(\theta)\left(1 - \frac{\|x\| - \|y\|}{max(\|x\|, \|y\|)}\right) \tag{6.11}$$

- S_6 is characterized by ellipsoidal constant-similarity surfaces, remembering the behavior of D_{eucl} and comparative results may be interesting. S_6 is defined by:

$$S_6(x,y) = 1 - \frac{(\|x\|^2 + \|y\|^2 - 2\|x\|\|y\|\cos(\theta))^{1/2}}{(\|x\|^2 + \|y\|^2 + 2\|x\|\|y\|\cos(\theta))^{1/2}} \tag{6.12}$$

In all definitions, $\|.\|$ means the Euclidean norm.

Strehl [140] put forward the behavior of the Jaccard similarity compared to the Euclidean distance and the Cosine similarity, which combines advantages of both as illustrated in Figure 6.6. According to that, the Euclidean distance is expected to group similar colors in a circled neighborhood while the Cosine similarity in line directions and the Jaccard similarity simultaneously in both. To handle absolute color differences, the Euclidean distance may be preferred and to handle varying colors (by definition of the RGB cube) the Cosine similarity may be, in contrast, better. The Jaccard similarity defined in Equation 6.13 may be expected to handle both cases.

$$S_7(x,y) = S_{Jac}(x,y) = \frac{xy}{\|x\|^2 + \|y\|^2 - xy} \tag{6.13}$$

Similarities named in this text from S_1 to S_7, with D_{eucl}, will be tested in Subsection 6.2.3 to understand the behavior of each metric and how to handle the aforementioned issues.

6.2.2 Noteworthy properties of angle-based similarities and complementarity with the Euclidean distance

Angle-based similarities and the Euclidean distance are complementary in various ways for color segmentation:

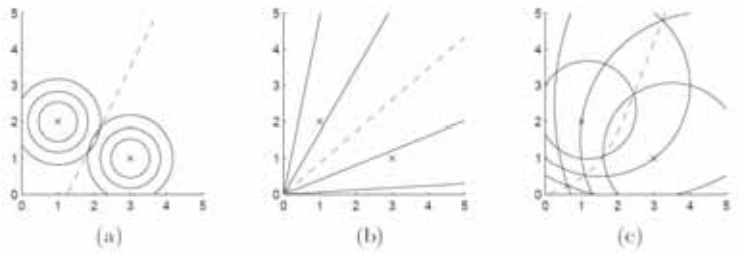


Figure 6.6: From [140], representation of iso-similarity surfaces for different metrics: the Euclidean distance (a), the Vector Angle similarity (b) and the Jaccard similarity (c). Two points are considered and three similarities values (0.25,0.5,0.75).

Hue representation Inside the RGB color space, a reliable and simple method to obtain hue information is through an angle-based similarity, which enables to dispose intensity and hue information in the same color space without complicated conversion.

Varying color characterization Similar colors have parallel orientations even when degraded with uneven lighting or by shiny material. In natural scene images, (slight) variations are a frequent occurrence within the same object of same color due to all sources of variations described in Chapter 2. Color can gradually change and by definition, an angle-based similarity can circumvent this issue.

Complementarity with the Euclidean distance An angle-based similarity represents chromaticity difference information whereas the Euclidean distance computes the intensity difference information. Their combination enables one to perform intensity-dependent segmentation directly from the RGB image in areas of different colors, and the other to perform intensity-invariant segmentation in regions of similar but not identical colors. Moreover, in a clustering process as displayed in Figure 6.7, we show the cluster definition done by both metrics for some natural scene images. From the RGB color space, the Euclidean distance separates pixels in the (R-G-B) view mostly in a horizontal (or radial) way with groups presenting quite the same volumes while an

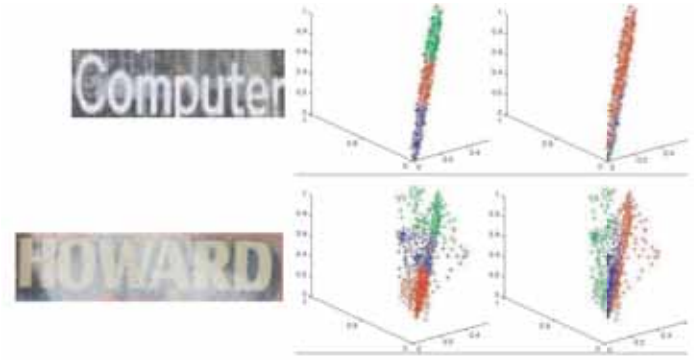


Figure 6.7: (R-G-B) view of clustering results done by the Euclidean distance and by an angle-based similarity (right) on initial images (left).

angle-based similarity does the same operation in a more vertical (or angular) way with groups presenting different sizes. These observations are quite logical due to the definition of each metric but really show a complementarity depending on colors in the image.

Both metrics enable one to handle a large number of degradations in a complementary fashion. Images presenting a strong contrast between text information and background are usually better segmented with the Euclidean distance. For the opposite case where images are corrupted by uneven lighting, shiny or curved surfaces, an angle-based similarity performs better. Due to the material, the angle of acquisition and the lighting, text colors vary gradually and can present strong differences inside a character. Since initially, colors inside the text were similar, it will be efficient to use an angle-based similarity because the angle of two colors remains small compared to the value of the Euclidean distance.

6.2.3 Evaluation of several metrics

This evaluation aims at proving that using only the RGB color space but with dedicated particular metrics, one can handle degradations of NS images without the need to design any new color spaces or combinations of existing ones. All metrics were devel-

Table 6.2: Precision, Recall and F-score measures for several metrics in a RGB-based k-means clustering framework.

	DB-ICDAR			DB-WWW		
	P	R	F	P	R	F
D_{eucl}	0.90	0.88	**0.89**	0.81	0.78	**0.79**
S_1	0.90	0.35	**0.50**	0.86	0.33	**0.48**
S_2	0.62	0.26	**0.37**	0.62	0.24	**0.35**
S_3	0.86	0.26	**0.40**	0.87	0.26	**0.40**
S_4	0.88	0.34	**0.49**	0.85	0.33	**0.48**
S_5	0.93	0.36	**0.52**	0.94	0.39	**0.55**
S_6	0.91	0.34	**0.50**	0.90	0.35	**0.50**
S_7	0.68	0.29	**0.41**	0.71	0.27	**0.39**

	DB-Sypole			**Average**
	P	R	F	F
D_{eucl}	0.91	0.89	**0.90**	**0.86**
S_1	0.84	0.21	**0.34**	**0.45**
S_2	0.63	0.24	**0.35**	**0.36**
S_3	0.84	0.19	**0.31**	**0.38**
S_4	0.82	0.20	**0.32**	**0.43**
S_5	0.90	0.19	**0.31**	**0.46**
S_6	0.90	0.22	**0.35**	**0.45**
S_7	0.68	0.19	**0.30**	**0.37**

oped using Matlab or in C language for computation time, detailed in some sections.

Table 6.2 details results of the eight tested metrics (from S_1 to S_7 added to D_{eucl}) inside a RGB-based k-means using the same measures as in Subsection 6.1.3, in terms of Precision, Recall and F-score.

Results are quite similar except for S_2 and S_7 where results are noisier and the number of correctly extracted characters decreases. S_2 is hence not convenient for text extraction, contrarily to the suggestion of Wesolkowski [158]. The accent must be put on cosine-based similarities for inclusion of hue information. S_6, whose aim was to combine advantages of the Euclidean distance and an angle-based similarity, gives satisfying results but the increase is not the one expected. Except S_2 and S_7, results of other similarities are quite consistent, with a small decrease for

DB-SYPOLE. Images of this database contain less uneven lighting, less shiny material and the use of angle-based similarities is therefore less relevant to handle varying colors. Based on the last column of Table 6.2, S_5 seems performing slightly better than other angle-based similarities.

As best metrics are very close in terms of results with F-score measures, the Hotelling Trace Criterion (HTC) [22] is used.

The HTC is a measure of class separability used in pattern recognition to find a set of linear features that optimally separate two classes of objects. Several scatter matrix-based metrics [153] may have been performed but we choose this one based simultaneously on cluster compactness and separability, respectively defined as $\mathcal{S}_w$ for the intra-cluster scatterness and as $\mathcal{S}_b$ for the inter-cluster scatterness.

The data set contains 3-dimensional samples $x = [x_1, x_2, x_3]$, as the RGB color space is used, to build C clusters (fixed to 3 for k-means). A combination between textual clusters may be useful and manual combination is performed to insure that results are independent of this point.

Each cluster c_i of the C clusters contains n_i samples with the total of n samples. The mean vector of each cluster is called μ_i and the total mean vector is μ. The scatter matrix, $\mathcal{S}_i$ for the i^{th} normalized cluster is:

$$\mathcal{S}_i = \frac{1}{n_i} \sum_{x \in c_i} (x - \mu_i)(x - \mu_i)^t \tag{6.14}$$

with $(x - \mu_i)^t$ meaning the transpose of the color vector minus the local mean μ_i.

The normalized intra-cluster scatter matrix ($\mathcal{S}_w$) and the normalized inter-cluster scatter matrix ($\mathcal{S}_b$) are computed as:

$$\mathcal{S}_w = \frac{1}{C} \sum_{i=1}^{C} \mathcal{S}_i, \; \mathcal{S}_b = \frac{1}{C} \sum_{i=1}^{C} (\mu_i - \mu)(\mu_i - \mu)^t \tag{6.15}$$

Tighter clusters (with smaller $\mathcal{S}_w$) that are far apart (with larger $\mathcal{S}_b$) are preferred. The HTC, $tr[\mathcal{S}_w^{-1}\mathcal{S}_b]$ computes the invariant measurements as it is also defined as:

Table 6.3: Scatter-based measures ($\mathcal{S}_m$) using the HTC measure for several metrics in a RGB-based k-means clustering framework.

	D_{eucl}	S_1	S_4	S_5	S_6
$\mathcal{S}_m$	11.55	27.02	26.73	24.42	20.77

$$tr[\mathcal{S}_w^{-1}\mathcal{S}_b] = \sum_{i=1}^{d} \lambda_i \tag{6.16}$$

where λ_i are the eigenvalues of $\mathcal{S}_w^{-1}\mathcal{S}_b$.

Separability computation is a theoretical measure for clustering quality taking no particular goal into account, such as object-driven segmentation.

Table 6.3 details results using scatter-based measures $\mathcal{S}_m$ for best metrics. S_1, S_4 and S_5 are the best discriminant metrics for NS images, meaning that intra-cluster similarity is small and inter-cluster one is large. However, it is interesting to note the weak discriminant power of D_{eucl}, which gives more satisfying results in terms of Precision, Recall and F-score. Hence, among angle-based similarities, we choose S_5 as the best one, because it has a satisfying discriminant measure and simultaneously gives the best F-score.

Best metrics, S_1, S_4, S_5 and S_6, have been tested in other color spaces. Nevertheless, results were not improved, showing the sufficiency of RGB. For color spaces using already polar coordinates to exploit hue information, an angle-based similarity was less relevant.

To add arguments to complementarity between the Euclidean distance and the best angle-based similarity for NS text extraction and simultaneously on all databases, D_{eucl} handles around 56% of images with poor results for S_5 while the inverse is true for 12%. It exists an overlap of similarly extracted text for 32% images only. These results are clearly dependent on databases. If images are clean, with good discrimination between foreground and background, angle-based similarities are useless.

It looks very necessary to efficiently combine the Euclidean distance and S_5, in another way of a single measure due to their very different definitions. A solution is proposed with the SMC algorithm, presented in Section 6.3.

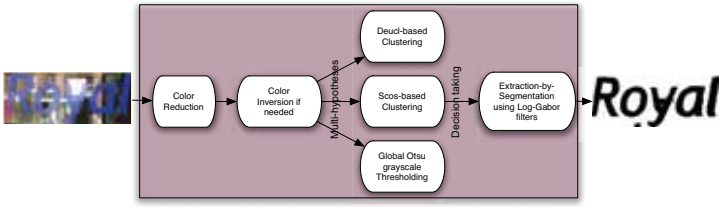


Figure 6.8: Steps of the SMC algorithm.

6.3 SMC - Selective Metric Clustering for Text Extraction

Text extraction is a challenging issue, made even more difficult in a NS context. Classical binarization algorithms on gray-scale images showed their limitations to handle NS degradations. Colors have to be taken into account and based on the preliminary studies of Chapter 6, we propose an algorithm that we call ***Selective Metric Clustering*** or ***SMC***. We perform a 3-means clustering algorithm using two metrics, the Euclidean distance and an angle-based similarity, equal to $1 - S_5$ in order to use the same k-means algorithm for both metrics. D_{eucl} is normalized between 0 and 1 like the angle-based similarity: '0' means colors are identical and '1' that they are totally different. The similarity will be called S_{cos} in the subsequent explanations. Moreover as stated in Chapter 2, intensity is paramount information to distinguish similar pixels of the same color but different intensities and SMC includes a gray-scale image, thresholded with a traditional global binarization to build a multi-hypothesis text extraction. Finally, as text is a meaningful object and as the chosen k-means clustering does not integrate spatial information, SMC opts for the proper text extraction by using clues of spatiality.

Figure 6.8 details steps of the SMC algorithm for text extraction and the following subsections detail each of these steps.

6.3.1 Color reduction and color inversion

First of all, the number of colors is drastically reduced in order to lower computational time. Considering properties of human vision and based on the interesting study of Pujol et al. [124], who

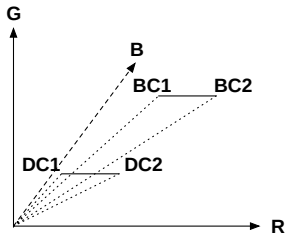


Figure 6.9: Larger angle between dark colors DC1 and DC2 than between bright colors BC1 and BC2, even if $d(BC2, BC1)$ equals to $d(DC2, DC1)$ with d, the Euclidean distance.

Figure 6.10: SMC extraction result without inversion (middle) and with inversion (right) on an initial image (left).

highlighted the short dynamic range of digital cameras compared to the human eye in some regions, we decided to represent each RGB channel with only 4 bits. It introduces very few perceptible visual degradation and does not affect the results. Hence the dimensionality of the color space is 16*16*16 and it represents the maximum number of colors.

Following this quick step, a color inversion is applied to make text always darker than background. The use of S_{cos} means exploitation of the angle between two color vectors relative to the origin of the RGB color space. This origin point corresponds to no illumination with R = 0, G = 0, B = 0. Hence, as illustrated in Figure 6.9, dark text on bright background is more appropriate to segment as angles with colors close to the origin have wider dynamics. Angle-based similarities add hue information inside RGB and similar to the hue drawbacks, it becomes unstable for small angles. Figure 6.10 shows the extraction result for an image with strong uneven lighting before and after inversion.

The choice of inversion is done after applying a quick global image thresholding with the well-known Otsu method [113], leading to the binarized image I_O; the maximum between black and

white pixels on the image I_O edges implies that text is brighter or darker than the background, assuming text is not mainly connected to edges. Equation 6.17 details the inversion decision. This assumption is right with off-the-shelf text detection algorithms.

$$\text{Inversion} = \text{true if (W+H) inferior to} \quad (6.17)$$
$$\sum_{i=0}^{H-1} I_O(i,0) + I_O(i,W-1) + \sum_{j=0}^{W-1} I_O(0,j) + I_O(H-1,j)$$

where W and H are the width and height of I_O. In Equation 6.17, convention is taken with background pixels equal to 1 and text ones equal to 0.

6.3.2 Utilization of a multi-hypothesis text extraction

SMC performs two clustering algorithms on the initial image with two metrics, D_{eucl} and S_{cos}, shown to be complementary in Section 6.2. Moreover, to alleviate effects of achromatic images and improve results of text extraction, we add intensity information with the thresholded (inverted) gray-scale image in the previous step. For pure achromatic images (meaning R = G = B), S_{cos} cannot build 3 clusters efficiently as all pixels are on the same diagonal in the RGB cube. The same phenomenon appears for non-pure achromatic images where it is rather difficult to separate colors efficiently. This drawback is also true in hue-based color spaces where hue is even not defined! As this third binarized image was already computed for another task, it does not add computational time. We obtain also three possible text extraction results of both metrics and the already-binarized gray-scale image. Before choosing the proper extraction, additional details on combination of clusters must be cleared out.

K-means clustering (with $k = 3$) applied on NS color images with two metrics forms 3 clusters for each one and one cluster is obviously a part of the background, another one is a part of the text and the third one is either text or background. For sharper results and hence better character recognition, it may be interesting to combine both textual clusters. In Wang et al. [156], combination was based on some texture features, to remove irrelevant clusters, and on a linear discriminant analysis. While in Thillou and Gosselin [144], the most probable textual cluster was defined

with a means of skeletonization by computing a global binarization on the initial image beforehand and the possible combination was finally based on differences of colors between the most textual cluster and the possible one.

In SMC, first of all, the background color is selected very easily and efficiently as being the color with the biggest rate of occurrences on the image edges. Next, we propose a new text validation measure $\mathcal{R}$ to find the most textual foreground cluster over the two remaining clusters. Based on properties of connected components of each cluster, spatial information is already added at this point to find the main textual cluster. $\mathcal{R}$ is based on the largest regularity of connected components of text compared to those of noise and background and is defined in Equation 6.18:

$$\mathcal{R} = \sum_{i}^{N} |area_i - \frac{1}{N}(\sum_{i}^{N} area_i)| \tag{6.18}$$

where N is the number of connected components and $area_i$ refers to the area of component i. This measure enables the computation of the variation in candidate areas. The main textual cluster is identified as the one having the smallest $\mathcal{R}$. If the third unknown cluster belongs to text, both textual clusters need to be merged. A new computation of $\mathcal{R}$ is performed considering the merging of both clusters. If $\mathcal{R}$ decreases, the fusion is processed. This method enables the merging of text of different colors in the same word for instance as regularity becomes better. Nevertheless, a text with each letter in a different color, for instance, could be only handled by increasing the number of clusters.

Finally, text extraction for each clustering metric is done. Figure 6.11 displays some results of this SMC step where text extraction is sometimes better using D_{eucl}, S_{cos} or the global binarization and hence showing the complementarity between the two metrics and the additional globally thresholded result.

With this multi-hypothesis text extraction, we may handle a very large range of NS images. The use of S_{cos} is preponderant, as illustrated in Figure 6.12 with some complex NS images which can not be better handled in a k-means framework. Some comparisons were done with the Euclidean distance and by increasing the number of clusters or with other color spaces. Angle-based similarities can extract text of very challenging NS images without additional effort and by keeping versatility for other NS images.

Figure 6.11: Display of complementarity between the three extraction hypotheses. From left to right: initial images, clustering result using D_{eucl}, clustering result using S_{cos}, globally thresholded result.

Figure 6.12: More extraction results using S_{cos} in a RGB-based k-means framework.

6.3.3 Extraction-by-segmentation

After computation of k-means with two different metrics, the choice between the three text extraction methods has to be done. A multi-hypothesis method has been shown by Chen [16] by varying the number of clusters in a GMM-based clustering and choosing the right segmentation with the final step of recognition. One drawback to this method is to keep several segmentations to process during subsequent steps and to increase the number of text areas to recognize. Moreover, recognition is logically an efficient step to choose the right segmentation, but in complex NS images, character segmentation or even denoising steps must be added, and no decision could be done before the final step of recognition; otherwise, recognition results may be erroneously considered bad. In SMC, we choose to intermingle consecutive steps to avoid this disadvantage and to add as much information as possible.

Color information is a very consistent clue for NS images. However the segmentation process, previously described in this

Figure 6.13: Same examples of Figure 6.11. From left to right: result of the absolute phase of the vertical log-Gabor filtering, after applying mask obtained by D_{eucl}-based extraction, after applying mask obtained by S_{cos}-based extraction and after applying mask obtained by global thresholding. More explanations about the use of log-Gabor filtering are given in Chapter 7.

chapter, does not make use of spatial information, which is quite necessary for object-driven segmentation and specifically text extraction. In order to extract characters properly, we exploit the same tool for character segmentation, detailed in depth in Chapter 7. We need to have spatial information to locate characters in the image, as well as needing the frequency information to use illumination variation to detect character edges. Hence, log-Gabor filters proposed by Field [35] are chosen for decision making, because they particularly fit well to NS images as explained in Section 7.2 and overviewed in Mancas-Thillou and Gosselin [91].

One important parameter for log-Gabor filters is the filter frequency. As we used them to enhance characters in a gray-scale image, we choose a frequency equal to the inverse of the rough thickness of characters, determined by the number of pixels of the extracted result and its skeleton. A simple ratio between these two latter values are computed and the inverse is the frequency of log-Gabor filters.

Figure 6.13 shows the result of the same three examples of Figure 6.11 after applying the mask of each segmentation performed previously. Results of log-Gabor filters present globally high responses to characters with this set frequency. High responses are illustrated by warmer colors, meaning that red characterizes highest responses and blue, lowest ones. Hence in order to efficiently choose the best extracted text result, we perform an average of pixel values inside each mask. The segmentation having the highest average is chosen as the final segmentation.

Table 6.4: Precision, Recall and F-score measures of text extraction performed by D_{eucl}-based k-means, S_{cos}-based k-means and the global Otsu thresholding [113].

	DB-ICDAR			DB-WWW		
	P	R	F	P	R	F
D_{eucl}	0.90	0.88	**0.89**	0.81	0.78	**0.79**
S_{cos}	0.93	0.36	**0.52**	0.94	0.39	**0.55**
Otsu [113]	0.88	0.76	**0.82**	0.89	0.80	**0.84**

	DB-Sypole			**Average**
	P	R	F	F
D_{eucl}	0.91	0.89	**0.90**	**0.86**
S_{cos}	0.90	0.19	**0.31**	**0.46**
Otsu [113]	0.94	0.92	**0.93**	**0.86**

6.3.4 SMC evaluation and results

Based on the evaluation part of Section 6.2 showing complementarity between the Euclidean distance and an angle-based similarity, Table 6.4 extends Precision and Recall results by adding the third hypothesis of text extraction with the thresholded gray-scale image. Moreover, one may note the insensitivity of the SMC method to inaccuracy of constrained textual areas. DB-ICDAR are manually segmented while the two other databases are automatically segmented using the publicly available algorithm of A. Chen [83]. To add more arguments to complementarity between these three extracted results, D_{eucl} performs better in 5 % images, while S_{cos} in 12% and the global thresholding in 9%. There is a larger overlap between D_{eucl} and the global thresholding which performs quite equally in 69% images.

To choose the right text extraction, we opt for log-Gabor filters by adding spatial information. In [90], we compared the performance of this method with the Silhouette technique, a measure of how well clusters are separated, to choose between the two metrics only. It can be logical to think that best text extraction results present the best separation between clusters. However, it is not always true because Silhouette performs well in 77.7 % images and the proposed method using spatial information performs well in 93.2 %, yielding an improvement of 19.9 %.

A few works deal with NS text extraction and we compare

Table 6.5: Comparison of Precision, Recall and F-score measures between Wolf's method [159] (W) , Garcia and Apostolidis's method [42] (G&A) and the proposed SMC method.

	DB-ICDAR			DB-WWW		
	P	R	F	P	R	F
W	0.35	0.19	**0.25**	0.32	0.16	**0.21**
G&A	0.66	0.57	**0.61**	0.60	0.52	**0.56**
SMC	0.95	0.91	**0.93**	0.91	0.86	**0.88**

	DB-Sypole			**Average**
	P	R	F	F
W	0.52	0.38	**0.44**	**0.30**
G&A	0.62	0.54	**0.58**	**0.58**
SMC	0.93	0.89	**0.91**	**0.91**

SMC, firstly, with solutions of Wolf [159] which designed an extended method of Sauvola and Pietikäinen [130] to extract text from NS images or videos, and then, with solutions of Garcia and Apostolidis [42] which used a k-means clustering in the HSV space with the Euclidean distance only. Combination of clusters in this last method has not been implemented and a perfect combination is assumed while the method is tested including our combination method. Algorithms were developed in Matlab for this comparison purpose. Results are presented in terms of Precision, Recall and F-score in Table 6.5.

The combination of two metrics in a clustering framework and a global thresholding has proven its efficiency compared to two recent and competing algorithms. Main errors of the method are due to low resolution still images and choice of the best result between the three hypotheses.

Finally, due to the explosion of use of camera phones or digital cameras and huge amount of images to process for text extraction, the algorithm needs to be relatively fast in order to provide satisfying results for frequent use. The text extraction algorithm runs in 0.61 seconds on average for databases on a PC with a Pentium M-1,7 GHz micro-processor. The source code for text extraction was developed in C language but could be optimized further such as with the triangle inequality technique [29] for k-means to reduce the number of distances to compute. Another

Figure 6.14: Error example of the selective metric-based clustering: initial color embossed image on left and the SMC result on right.

optimization method could be to compute D_{eucl} and S_{cos} simultaneously by taking advantage of some calculations between both.

6.4 Conclusion of the Selective Metric Clustering Technique

In this chapter, the SMC algorithm has been proposed based on a multi-hypothesis text extraction by selecting either the right clustering metric or the dual information between color and illumination, using log-Gabor filters. Several points have been detailed:

- Superiority of metrics over color spaces in a clustering framework inside a general NS context. Angle-based similarities have overcome any other color spaces to handle complex NS images, meaning mainly images with complex backgrounds and uneven lighting.

- Complementarity between the Euclidean distance and angle-based similarities in a k-means method to handle a very large set of NS images with respect to image formation issues.

- Addition of spatial and luminance information to choose the best text extraction to provide to recognition. To circumvent NS challenges, text extraction was intermingled with the subsequent step of character segmentation.

- Very encouraging results detailed in Subsection 6.3.4 in terms of Precision, Recall and F-score, comparison with other state-of-the-art algorithms, and while keeping a reasonable computation time.

The selective metric-based clustering is aimed at being versatile and results we have provided show that it is. Nevertheless,

SMC mainly uses color information and one drawback of the system is for natural scene images having embossed characters. In this case, the foreground and background have the same color imparting partial shadows around characters due to the relief but not enough to discriminately separate the textual foreground from the background as displayed in Figure 6.14. Gray-level information with the simultaneous use of a priori information on shadows and character properties could be a solution to handle these cases. Nevertheless, it may be relevant to note that a robust OCR may also give satisfying results without any modifications of the algorithm.

— CHAPTER 7 —

Unit-based Segmentation

This chapter deals with segmentation of text areas into specific units, such as lines, words and characters. In commercial OCR systems, this process is usually included and is quite successful except for severely degraded characters, strongly broken or tightly connected ones where recognition rates drastically drop. Incorrect segmentations due to perspective, for example, may even lead to no recognition at all. Usually, NS text, handled in literature, is well separated due to their reading goal. However, complex NS images with low resolution, perspective or wavy surfaces present challenges and unit-based segmentation has recently become a point-of-interest to circumvent recognition errors. Hence, we describe a fast and simple line and word segmentation method in Section 7.1 and an innovative and robust character segmentation method using log-Gabor filters in Section 7.2.

7.1 Line and Word Segmentation

NS images may present several words but usually only a few lines if we cite street names or book titles. Nevertheless, colorful magazine headlines or abstracts on book covers or even camera-based documents such as restaurant menus may have several lines. Line and word segmentation are usually not considered as difficult for NS images but present interesting challenges for skewed text areas; as such we present very fast and intuitive algorithms.

To perform a low-level segmentation into lines, words and characters, contrarily to high-level meaning structure layout and anal-

ysis such as paragraphs, titles and so on, the system may be top-down or bottom-up. For the top-down analysis, a page is segmented from large components to smaller subcomponents, also from lines to characters. For bottom-up analysis, connected components are merged into characters, then words and finally, text lines. Both methods may also be combined for a hybrid analysis, which is more robust due to the use of more information about text on each sub-step. To describe the hybrid process, we segment text into characters, then into lines, back into refined characters with supplementary information and finally into words.

7.1.1 Line segmentation

Segmentation into lines is an old topic and the two main and successful methods are either the vertical projection profile or the Hough transform [53]. The first one is a histogram of the number of text pixels accumulated along text lines and projected vertically. The projection profile has maximum-height peaks for text and valleys for inter-line spacing. It is quite sensitive to noise and skewed lines. The second method maps each point in the original (x,y) plane to all points in the (r,θ) Hough plane of possible lines through (x,y) plane with slope θ and distance from origin r. This method performs well on skewed text and may also simultaneously deskew it with the knowledge of θ value but it is on the other side computationally quite expensive. Deeper explanations of the two algorithms may be found in [110].

Connected components coming from the text extraction step to perform the deviation measure $\mathcal{R}$ in Subsection 6.3.2 are already computed with general properties, such as height of characters h_{char}. On the strict bounding box of the text area, we define the approximate number of lines N_l by:

$$N_l = floor\left(1 + \frac{(h_{text} - \mu(h_{char})/2)}{\mu(h_{char}) * 3/2}\right) \tag{7.1}$$

where h_{text} is the height of the text area, $\mu(x)$ is the average of x on all characters and $floor(x)$ is the largest integral value less or equal to x.

All y-coordinates of character centroids are then clustered with the k-means algorithm, k being equal to N_l. Figure 7.1 explains the concept of line segmentation. For databases, no error occurs in line segmentation.

Figure 7.1: Illustration of line segmentation for skewed text: three clusters based on the y-coordinates of connected components.

For strongly skewed lines, a fast deskewing is required based on the height of the text bounding box. The first text pixel of the first row of the tightest bounding box is detected and if its position is before the middle of the image width, the skew angle is negative; otherwise it is positive. A first rotation of 1° is computed in the determined direction. If the bounding box is shorter in height than the previous one, successive rotations are performed until the bounding box becomes higher meaning that the skew angle was larger than 1°.

More complex algorithms, such as detection of horizontal or vertical writing for multi-language documents, may be designed but are not necessary for NS images and out of scope of this text.

7.1.2 Word segmentation

Word segmentation, contrarily to line segmentation useful for better character recognition, is a crucial step for text understanding after recognition, such as by speech synthesis. A natural linguistic parser is always part of a text-to-speech algorithm and it is important to identify words for a proper pronunciation as explained in the example:

Ex: in French, the phonetic transcription can be different, depending on word segmentations:

" les tas " $\Rightarrow$ [l e t a]
" lestas " $\Rightarrow$ [l ϵ s t a]

In Latin alphabets, the inter-words distance D_{IW} is larger than the one of inter-characters D_{IC}. We compute word segmentation by identifying word separations by all distances superior to $std(D_{IC})+mean(D_{IC})$ with $std(.)$ and $mean(.)$, respectively standard deviation and mean of inter-character distances in a given line.

This step occurs after the refined character segmentation in order to have more correct calculations based on characters and spaces between characters.

For this step, we use a simple statistic method. Some errors may occur when a few words are present with distances between words varying due to different fonts or perspective. Nevertheless, this algorithm is robust when run against text areas presenting only one word, which is quite frequent in NS images or after text detection algorithms, which usually oversegment lines.

Finally, this rule basically bends to oversegmentation more than subsegmentation, which may be more easily handled by the recognition correction step using linguistic information and finite state machines as described in Chapter 8. Complex word segmenters are, in any case, based on addition of linguistic information such as Wang's one [155], which is designed for Latin alphabet as well and uses word entropy in order to segment them properly.

7.2 Character Segmentation using Log-Gabor Filters

7.2.1 Is character segmentation still useful?

The first character segmentation algorithms, developed for typewritten characters, appeared more than forty years ago to separate each character individually, in order to subsequently feed into OCR. Later, these techniques have been extended to segmentation of cursive writing for handwritten text.

Main techniques for typewritten characters are categorized into three groups:

- Image-based methods are mainly issued from projection analysis, either vertical projection of text pixels leading to a histogram with valleys representing vertical separation between characters, or differencing measure after column ANDing (relative to the logical AND operation), or the

"Caliper" distance, which is the distance between the uppermost and bottommost pixels in each column meaning that smallest distances are tentative segmentation places, as experienced in camera-based document processing [143]. These methods imply vertical separation only, which is not convenient at all for strongly joined characters or skewed and italic ones where parts of a character infringe on the space occupied by the next one. Moreover projection analysis is very sensitive to noise and hence to text extraction errors.

- Recognition-based methods use a sliding window of variable width to provide sequences of hypothetical segmentation locations which are confirmed or refuted by character recognition. These techniques also give only vertical separations and need robust OCR to reject or accept all possible segmentations, which are quite numerous, even for a single word! Markov approaches, used most often for handwritten text, represent letter-to-letter variations of the language to validate tentative segmentations after dissection into individual characters. Segmentation cuts can also be determined by a particular recognizer, as in Bae et al. [5], where all possible cuts have been previously learnt. It is obvious that this latter method can not fit NS images regarding the diversity of images and ways of character connect.

- Hybrid methods mainly encompass oversegmentation methods. A word is dissected into its smallest possible components and recognition is based on these units to individually recompose the characters one at a time. Droettboom [25], for historical printed documents, used graph theory to rejoin components together. Another perfect example is the oversegmentation of Lecun et al. [72] which builds a segmentation graph associated with a convolutional network, which is a robust recognizer, and provides very impressive results. They are particularly well suited for joined and broken characters and segmentation results are not only vertical as based on small components. Nevertheless, oversegmentation techniques need a dedicated recognizer based on unit features.

Details of all classical typewritten character segmentation are out of scope of this text and enlightening surveys may be found in Casey et al. [14] and Fujisawa et al. [40].

Recently, recognition algorithms, free of segmentation, have appeared to circumvent errors and difficulties of this step. But they all needed to assume a priori information on text such as Fang [31] and Kim [63], which is not in favor of versatility. They based their methods on the presence of several occurrences of the same character in a text area and by adding linguistic information, they deciphered the text area with characters similar to others. A consistent font over the text area is needed in this case. The largest category of recognizers, free of segmentation, is the word recognizers one, extensively used in handwritten documents. Nevertheless in NS images where words are totally unknown and even not in a dictionary like numbers, brands and so on, these solutions are implausible.

In a framework as the one detailed in Chapter 4, NS images need robust character segmentation since not all aforementioned methods are suitable, and off-the-shelf OCR using them lead to too many recognition errors. A gap between complex NS images, as the one displayed in Figure 3.4 in Subsection 3.2.2, and character recognition has to be filled to extend applications and use of NS images in daily life.

A few papers deal with character segmentation into individual components for NS images. This recent field focused more on text detection and localization and more recently text extraction, with images sometimes having difficult-to-extract text but largely spaced characters. For WWW images, Karatzas and Antonacopoulos [61] used a proximity measure to add information for text extraction to avoid connecting or breaking components. Hence, the number of wrong segmentations was reduced but no solution was given to correct the issue. Myers et al. [104] corrected perspective on NS images such as street signs to also reduce the number of connected characters and increase recognition rates. Chen [16] and Kim et al. [67] added spatial information to segment characters into individual components. The first method uses MRF-based text extraction to reduce connected characters and adds a gray-scale constraint to remove spurious parts between components. It assumes that gray-level information inside a NS character is nearly uniform. The second one exploits block information using character structural information, and a confidence factor based on recognition probability enables the verification of character segmentation of camera documents, after a spatial split-and-merge technique.

Based on these considerations, a NS character segmenter is needed to increase NS character recognition and has to be robust:

- against already individual characters, broken and joined ones
- against unknown fonts, italic characters or with perspective (in a reasonable degree).

A very innovative solution, using log-Gabor filters and the recognition step that follows in a hybrid method, is fundamentally different from existing ones, and is presented in Subsection 7.2.3 after focusing on properties of these filters in Subsection 7.2.2.

7.2.2 Why are log-Gabor filters appropriate for NS character segmentation?

Character segmentation in NS images obviously needs text properties and gray-level information to complement the color information exploited in text extraction in Chapter 6. Hence simultaneous spatial and directional information (for character separation location) and frequency information (gray-level variation to detect cuts) are required. Gabor filters are a traditional choice to address this issue: they are cosine-like filters having a given direction and modulated by a Gaussian window. They have been extensively used to characterize texture, and more specifically in this context, to detect and localize text into an image. In this aim, Gabor filters are quite time consuming because several directions and frequencies must be used to handle the variability in character sizes and orientations. Moreover, Gabor filters present limitations: large bandwidth filters induce a significant continuous component and only a maximum bandwidth of 1 octave could be designed. Field [35] proposed an alternative function called log-Gabor which lets us choose a larger bandwidth without producing a continuous component. Moreover, he suggested that natural images are better coded by filters that have a Gaussian transfer function on a logarithmic frequency scale, by showing that their spectrum statistically falls off at approximately $1/f$, which corresponds well to where the log-Gabor filter spectrum falls off on a linear scale. Figure 7.2 displays the shape of log-Gabor functions at the same frequency but with bandwidth varying from 2 to 8 octaves. Log-Gabor functions have the same appearance as Gabor functions for

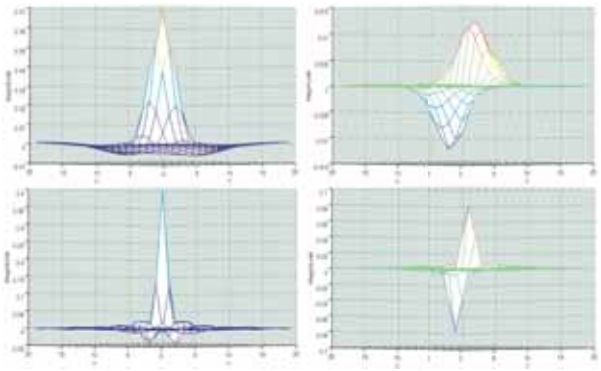


Figure 7.2: From top to bottom: even (left) and odd (right) log-Gabor filters with a bandwidth of 2 octaves and even (left) and odd (right) log-Gabor filters with a bandwidth of 8 octaves. In the spatial domain, the possibility of sharpening the filters is highlighted.

bandwidths less than one octave. The possibility of sharpening the filters is highlighted.

Log-Gabor filters in the frequency domain can be defined in polar coordinates by $H(f,\theta) = H_f \times H_\theta$ where H_f is the radial component and H_θ, the angular one:

$$H(f,\theta) = \exp\left\{\frac{-[ln(f/f_0)]^2}{2[ln(\sigma_f/f_0)]^2}\right\} \times \exp\left\{\frac{-(\theta-\theta_0)^2}{2\sigma_\theta^2}\right\} \qquad (7.2)$$

where f_0 is the central frequency, θ_0 is the filter direction, σ_f is the standard deviation of the radial components of the Gaussian describing the filter and is used to define the radial bandwidth B in octaves with $B = 2\sqrt{2/ln(2)} * |ln(\sigma_f/f_0)|$ and σ_θ is the standard deviation of the angular part of the Gaussian and enables the definition of the angular bandwidth $\Delta\Omega = 2\sigma_\theta\sqrt{2ln(2)}$.

As we are looking for vertical separation between characters, we only use two directions for the filter: the horizontal and the vertical ones. Hence, for each directional filter, we have a fixed angular bandwidth of $\Delta\Omega = \Pi/2$. Log-Gabor filters are not really strict with directions and defining only two directions enables the handling of italic and/or misaligned characters. For highly misaligned characters, the number of directions could be increased to

Figure 7.3: Correction of misestimated thickness with varying bandwidth. From left to right: original image, segmentation with misestimated thickness, segmentation with the same thickness corrected by a larger bandwidth.

handle this additional degradation, but it is important to mention that the angular bandwidth will become narrower and hence more selective.

Only two parameters remain to be defined, f_0 and σ_f, which are used to compute the radial bandwidth. The central frequency f_0 is used to handle gray level variations to detect separation between characters. The spatial extent of characters is their thickness that we consider as the wavelength of 'characters', hence it is logical to get a central frequency close to the inverse of the thickness of characters to get those variations. However, the measurement of character thickness may not be very accurate depending on the presence of degradations. In order to handle all kinds of degradations, we compensate for inaccurate thickness estimation with the second parameter σ_f. If the thickness of characters is not consistent inside a character such as in Figure 7.3, some character parts can be removed permanently. In this case, by increasing the bandwidth, we can support the variability in the thickness of characters with a 'larger' filter. Moreover, sometimes with very degraded or close characters, the thickness is very difficult to estimate and the filter must be very sharp to get each small variation in the gray level values such as in Figure 7.4.

In Mancas-Thillou and Gosselin [93], we segmented characters individually by using log-Gabor filters in a static way, presenting problems with noisy characters or inconsistent character thicknesses. Hence, as degradations and conditions of frequency estimation are quite unexpected, we chose the bandwidth filter in a dynamic way using recognition results. In the following subsection, we detail our method and how each parameter is estimated.

Figure 7.4: Impact of varying log-Gabor bandwidth for character segmentation. Original image (top left), binary version (top right), segmentation with large bandwidth (bottom left), segmentation with narrow bandwidth (bottom right).

7.2.3 Character segmentation-by-recognition

Based on the binarization of the detected area, which is available with the proposed SMC algorithm in Chapter 6, the character segmentation may now be performed on gray-level images.

Frequency estimation To define frequency, a classical way is to use a "wavelet-like" method. This means trying out several frequencies to get a good result for one of them. This method is time consuming due to several convolutions with multiple frequency filters and the number of computations rose to the power of two with the second parameter.

Text embedded in natural scene images presents a quite consistent wavelength, which is very different from the background. For the filter, we decided to use a wavelength related to the average of the character thicknesses. This is computed by using the ratio between the number of pixels of the first mask obtained by the SMC method and its skeleton as shown in Figure 7.5. Hence, the central frequency f_0 can be estimated approximately by:

$$f_0 = \sum_{i,j} skeleton(i,j) / \sum_{i,j} mask(i,j) \qquad (7.3)$$

Bandwidth estimation-by-recognition Due to the large variation in NS character fonts and sizes, the bandwidth has to be chosen dynamically. As objects to be segmented are text, we can use segmentation-by-recognition to choose the convenient bandwidth. We fix the initial and final values for

Figure 7.5: Visualization of thickness estimation. Left: original image, right: *mask* (white) obtained by the SMC method and its *skeleton* image (red).

Figure 7.6: Log-Gabor filtering results for each filter property. From top to down: phase of the horizontal filter, phase of the vertical filter, magnitude of the vertical filter and absolute phase of the vertical filter.

the bandwidth estimation. From approximately 2 octaves to approximately 8 octaves, which makes σ_f/f_0 vary with a step of 0.1 (from 0.1 to 0.6), we process six filters and provide the result to an OCR engine.

The result is composed of the vertical filter only as the character separation is mainly vertical. Moreover, in the output, only the phase of the filter will be exploited. As the text and background information have different wavelengths, the phase contains much more information than magnitude. Moreover, local variation issued from the initial separation between characters induces a phase difference. The latter one contains the gray-level information while the phase shows a local map which makes a good separation between the background and the textual information; this intermediate result is then multiplied by the first mask from text extraction to remove possible noise around characters as displayed in Figure 7.6.

Figure 7.7: Phase of the vertical filter multiplied by the *mask* issued from the text extraction (top) and result after global thresholding (bottom left). Improvement is obvious from the binary version (bottom right).

As shown in Figure 7.7 after filter convolution, characters have mainly low intensities and higher background intensities. In order to remove spurious parts between characters and to remain parameter-free, we use a global Otsu thresholding [113], which automatically chooses the threshold to minimize the intra-class variance of the thresholded black and white pixels. With the use of the absolute phase of the vertical filter, only one threshold needs to be determined. After this step, we get a result, such as the one shown at the bottom of Figure 7.7, to choose the bandwidth for filters.

We use a home-made OCR algorithm composed of a multilayer perceptron with geometrical features to recognize characters, which is trained by a separate data set and is used to assess how well characters are segmented. Detailed explanations about this in-house OCR are provided in Chapter 8. After applying log-Gabor filters, connected components (mostly characters) are given as inputs to OCR. Figure 7.8 shows two examples with varying bandwidths and results from recognition, which enables us to make the right decision for the bandwidth estimation. Recognition rates for each character or assumed character are averaged and the maximum score enables the choice of the bandwidth. The first example is an image with little contrast between characters and background, and the second one presents misaligned and slanted text, highlighting the robustness of the algorithm. This estimation needs six straightforward filters with only one frequency which enables the use of log-Gabor filters for character segmentation in a low-resource context.

Figure 7.8: Character segmentation-by-recognition using recognition rates. 1^{st} and 3^{rd} columns: character segmentation with bandwidth varying from 2 to 8 octaves, 2^{nd} and 4^{th} columns: OCR results with average recognition rate, based on retrieved connected components, shown in blue. For the first example, the estimated thickness is 13 pixels and for the second one, 17 pixels, leading respectively to a frequency of 0.48 and 0.37 cycles/degree.

More examples are given in Figure 7.9 to appreciate performance of this proposed character segmentation based on log-Gabor filters. From top, the third example is composed of severely joined characters and the result after segmentation is very satisfying. Between 'i' and 'n' of the word 'smokin', the connection is still present but the recognition is now successful even with off-the-shelf OCR including traditional segmentation. The last example illustrates an original image with characters of two different major colors (yellow and white) and a yellow and blue background. Based on the combination of clusters, the 'M' of the word 'Memorex' has been reconstituted but simultaneously with some parts of

background. Nevertheless, the yellow background information has a different intensity and frequency than the 'x' character, leading to a successful segmentation.

Figure 7.9: More character segmentation examples. From left to right: original image, SMC-based binary version and result after character segmentation.

Even if in NS images, broken characters are rare due to the relatively large thickness of characters whose aim is to be read, it may be useful to have solutions for handling them. To recompose parts of a single character, we proposed in [92] an algorithm using log-Gabor filters as well. It enables the correction of already broken characters (particular fonts or text extraction errors) and new broken characters due to recognition failures. The bandwidth is fixed and the frequency estimation is refined by an iterative log-Gabor convolution.

Figure 7.10 details the algorithm where steps 1 to 4 are displayed. Step 1 computes the initial character segmentation, and in order to recompose parts of broken characters, components are considered by pairs in Step 2 by applying an iterative dilatation between them until they become a single component. Step 3 is a second iteration of the log-Gabor filtering. For this iteration if the two fused objects were really two characters, they will be separated a second time. If they were in reality two parts of the same character as for the 'U' or 'X', they will remain fused. The difference is that the frequency is recomputed for Step 3 to refine it and is hence more accurate. This fact helps to correct some of the false alarms as the 'X' which was incorrectly split during the first log-Gabor iteration because of a wrong estimation of the frequency, for example.

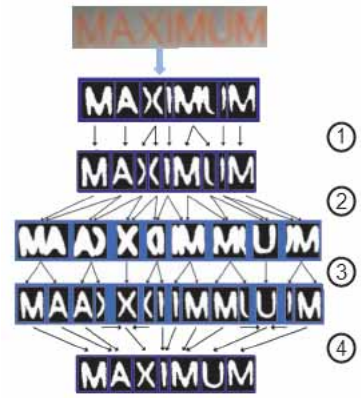


Figure 7.10: Schema of correction of broken characters. From top to bottom: initial image, text extraction result, first log-Gabor iteration (Step 1), objects pair grouping by dilatation (Step 2), second log-Gabor iteration (Step 3), and final character selection (Step 4).

Finally the fourth step consists of taking the decision on the last character segmentation. If an object is segmented twice in the same way, it means that it contains a unique character. If this is not the case, the character was broken and we have to fuse the two objects to get the entire character. Excepting the first and last objects, 'A', 'I', and 'M' are segmented twice in both pairs of objects: there were correctly segmented and they are single characters. For 'X' and 'U', the objects are not the same twice. In this case we obtain three objects: two lateral ones from the broken characters, and a central one which contains the objects fused into a single character. We shall choose the fused object in order to eliminate the broken parts. At this step we can add a validation to have more robust results: by fusing the lateral broken objects, we should obtain the same central object containing the fused character. If it is not the case, hence an error occurred at the third step: therefore, the final result will be a word with one (or more) lacking character(s).

The convolution of text extraction results with log-Gabor filters has several goals: to choose the better extracted text, to segment characters into individual parts and also to fuse broken characters by validating or not previous outputs. Log-Gabor fil-

ters give a large set of applications in NS images with a large modularity and very satisfying results as detailed in Subsection 7.2.4.

7.2.4 Evaluation

Similarly from the beginning of this text, all results are computed with databases mentioned in Chapter 4, except DB-VideosPDA. To compute log-Gabor filtering, we use the Kovesi' toolbox [69] in Matlab. The home-made OCR, which is useful to choose the right bandwidth, has been extended in *C* language from a version of Gosselin [48]. The "Caliper" distance and evaluation measures have been developed in Matlab.

In Table 7.1, comparisons are done between the behavior of an efficient commercial OCR (detailed in Chapter 4) against initial images without any processing, after the SMC-based text extraction without character segmentation, after a classical "Caliper" distance-based segmentation and after the log-Gabor-based segmentation-by-recognition to show the efficiency and necessity of this latter method to improve recognition results. Error rates are computed using the Levenshtein distance [73] between the ground truth and the resulting text. The Levenshtein distance or *edit distance* between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single character. Equal weights for each operation are employed in our computation. Error rates are then computed by dividing with the number of characters. By using the Levenshtein distance, some error rates for a word may be superior to 1, but it is useful to penalize broken characters. Tests have been computed on 10% of the databases due to the impossible automatic processing with a commercial OCR.

DB-ICDAR and DB-WWW contain more complex images with closed characters which may become joined characters after applying the SMC method. "Caliper-based" segmentation into individual components give better results for DB-Sypole, which is composed of more traditional and clean scenes. On DB-ICDAR, it even gives worse results than without segmentation. It is mainly due to the number of broken characters which increase. Log-Gabor segmentation drastically decreases error rates.

In Table 7.2, we compare the number of broken and connected characters between dynamic log-Gabor-based segmentation and

Table 7.1: Usefulness of character segmentation in natural scene images stated from recognition error rates with a commercial OCR.

Error rates	color images	SMC-based images	"SMC + Caliper" images	"SMC + Log-Gabor" images
DB-ICDAR	71%	40%	43%	19%
DB-Sypole	37%	22%	19%	21%
DB-WWW	65%	38%	35%	17%
Average	**58%**	**33%**	**32%**	**19%**

the static one with a fixed bandwidth of 1.7, which is the best value for databases, as mentioned in [89]. Hence, the number of joined or broken characters decreases with the dynamic character segmentation-by-recognition. DB-WWW is the database which contains most of broken characters due to the low resolution of images. Effort must be brought for LR web images.

Table 7.2: Impact of segmentation-by-recognition.

	Touching characters	Broken characters
fixed bandwidth	9.6 %	14.8 %
seg.-by- reco.	5.3 %	7.3 %
improvement	55.5 %	49.3 %

Finally, in this proposed character segmentation, the bandwidth is estimated with the recognition step and we compute the efficiency rate of this decision. Some erroneous choices could be made due to the majority vote on the whole text and the decision is correctly taken in 98.1% of images. Errors are mostly avoided with this character segmentation-by-recognition as each decision is checked with other steps dynamically. Main errors are either due to the OCR engine with much degraded characters or to the presence of thin characters. As log-Gabor filters exploit intensity information to accurately segment characters into individual components, if characters are too thin, they will be easy to break in several pieces of characters, leading to erroneous recognition.

Another benefit of log-Gabor filters in text understanding needs to be mentioned: the possibility to correct erroneously extracted text. In Figure 7.11, an example of this case is dis-

played. Between the three extractions, the first one is obviously the best one but some errors may occur in the choice of the best segmentation or even in the previous step of cluster combination for example. With the simultaneous combination of spatial and gray-level information, log-Gabor filters may enhance results to reduce the impact of errors. Due to an erroneous choice between the three binary hypotheses, the word 'point' is more difficult to segment (even if the other solutions are not really satisfying), but log-Gabor based segmentation is impressive again, leading to a total separation between characters. Even if some details have been lost (on the 't' of the word 'point'), the recognition is now successful.

Figure 7.11: Denoising impact of log-Gabor segmentation. From top to bottom: original image, three binary hypotheses from SMC algorithm (D_{eucl}-based (left), S_{cos}-based (middle), global thresholded (right)), binary result chosen and segmentation result using log-Gabor filtering.

Some deeper comparisons have been done with a recent method from Gatos et al. [43], who used the same public database. Their text extraction is based on a gray-scale adaptive thresholding and they proposed to recombine characters components based on several rules to avoid too many joined characters. In their evaluations, they included the step of text detection, which adds some errors. Hence, in order to compare similar methods, we do not use the manual text locating method but the public A.Chen's one [83] on the same images. Moreover, we use the same evaluation method being the Levenshtein distance [73]. Results

are displayed in Table 7.3. Improvement from Gatos et al. [43] may be observed with an error rate decreasing of around 43%. Note that 51% of errors come from text locating part and 8% from OCR (the home-made OCR recognizes mistakes of the used OCR). Errors of the proposed method mainly come from the choice between the three SMC-based hypotheses.

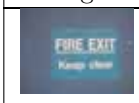
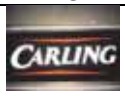
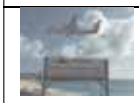
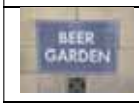
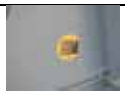
7.3 Conclusion of the Log-Gabor-based Character Segmentation

In this chapter, we propose NS character segmentation-by-recognition based on log-Gabor filters whose some parameters are defined dynamically. This algorithm fulfills requirements, established in Subsection 7.2.1, and gives interesting results under various aspects:

- No assumption on characters fonts, sizes or skew is done
- Characters are segmented with not only vertical separations but cuts following the character profile, leading to increased recognition rates
- Touching and broken characters are handled
- The algorithm is made more robust by using additional information with the consecutive step of character recognition
- Satisfying results in terms of recognition rates as well as the number of connected and broken characters are obtained
- The algorithm may even handle some errors of the previous text extraction step.

To conclude, log-Gabor filters are very modular and efficient tools to segment NS characters into individual and understandable components.

Table 7.3: Comparison of OCR results between the use of an OCR alone (O), Gatos et al.'s method [43] (G) and the proposed method ($\mathcal{M}$). Evaluation is based on Levenshtein distance [73] from the ground truth.

NS images	O	G	$\mathcal{M}$	NS images	O	G	$\mathcal{M}$
	21	0	0		2	1	1
	25	18	6		2	2	0
	5	4	0		32	3	4
	2	2	1		2	0	0
	3	3	4		39	18	19
	1	1	0		10	1	1
	0	0	0		10	10	0
	2	1	1		6	3	0
	0	0	0		38	16	10
TOTAL	–	–	–	TOTAL	201	83	47

— CHAPTER 8 —

Considerations on NS Character Recognition and Correction

From resolution enhancement to unit-based character segmentation, the main goal was to improve extracted text in order to finally increase recognition rates.

Character recognition is made easier with previous robust preprocessing steps. Hence, in this text, the objective was to provide high-quality extracted text in order to exploit off-the-shelf OCR. Nevertheless, NS character recognition, faced with the very large diversity of images without any a priori information, needs suitable conditions to work properly, such as a huge and significative training database or completion of the frequent NS cut words. Instead of focusing on convenient selected features for the recognizer, optimization of learning and so on, which in itself is enough to cover an entire thesis, we shall mention considerations on NS character recognition in Section 8.1 and on recognition-by-correction with the use of efficient and lightweight finite state machines in Section 8.2.

8.1 NS Character Recognition

8.1.1 What is done in NS character recognition?

From the origin of OCR in 1870 when Carey invented the retina scanner being an image transmission using a mosaic of photocells, through the real start in 1950 within the business world, to recent

breakthroughs in online and offline recognition, much effort have been done to decrease computation time and increase recognition rates. With fields varying from typewritten characters to historical handwritten ones, OCR needed dedicated algorithms for each category. Nevertheless, some progresses made OCR work efficiently for clean, binarized typewritten characters and recognition of NS characters within the framework aims at exploiting standard OCR.

Main character recognizers use:

Feature extraction: The objective of feature extraction is to capture the essential characteristics of the patterns. Building a feature vector is probably the single factor in achieving high recognition performance in OCR. Conventional features characterize distribution of points, transformations and series expansions of structural analysis such as moments, n-tuple, characteristic loci and so on, for either binary images, skeletons (thinned characters) or gray-scale images.

Feature selection: Several features may be redundant creating overhead computation or in the worst case scenario, confusion if some features bring contrary information for the same character. Main feature selection methods are principal components analysis, exhaustive search, branch-and-bound, iterative selections and so on [123].

Classification: Obviously, a classifier is required based on features or pixels themselves. Among template-matching, k-Nearest Neighbor (kNN) or other kinds of classifiers, statistical supervised recognizers, such as hidden Markov models or neural networks, are preferred and have proven real efficiency in OCR.

The reader must understand that a state-of-the-art character recognition system may be huge if detailed, which is out of scope of this text, therefore the reader is referred to the excellent survey of Jain et al. [57], on this topic.

To focus on NS character recognition, main recent papers deal with gray-level characters to handle degradations and low resolution of acquisition. The idea is therefore to build efficient recognizers against some issues without improving characters beforehand.

For WWW images, Zhou et al. [167], first extracted characters by color clustering and then converted the characters' colors into

gray-scale. The main color receives the value of 255 and the other ones are set to differences from the representative color. The character shape is then treated as a 3D surface and a polynomial surface fitting method (Legendre polynomial basis) is used as feature extractor and a basic character-to-class Euclidean distance is used to recognize characters. For NS text, Zhang et al. [165] exploited also gray-scale images after intensity normalization with Gabor-based features in the context of Chinese sign recognition. They performed feature selection with a linear discriminant analysis to build a space as discriminate as possible. Finally the classification is solved with kNN. Yokobayashi and Wakahara [163], after local binarization on CMY color planes, performed affine-invariant gray-scale recognition as well as using global affine transformation correlation, a particular template matching technique. To circumvent poor binarization of mobile phone camera-based images, Sun et al. [141] extracted features based on dual eigenspace. A sub-pixel gray-scale normalization is used first, then, the recognition is done by comparing similarities of features with synthetic generated patterns. Usual training database to recognize characters are not noisy and high resolution, hence they created a degraded dictionary using a video degradation model based on perspective transformation.

8.1.2 Description of the exploited recognition system

To perform segmentation-by-recognition in Chapter 7, we use an extended version of classifier from Gosselin [48], based on geometrical features and a multi-layer perceptron (MLP).

Briefly, an MLP is a network of simple processing units arranged into a hierarchical model of layers. The units (*neurons* or *nodes*) in the first (*input*) layer are connected to nodes in the subsequent layer(s) (*hidden layers*), to the final (*output*) layer. Numerical input vectors of patterns are presented at the input layer, and activity flows through the network to the output layer. Connections have a numerical weight value associated with them, and the signal transmitted via a connection is multiplied by the weight value. Each unit computes some function of the sum of its weighted inputs, and transmits the result through its output connections by comparing with a threshold, as shown in Figure 8.1. This kind of activation function is generally the **sigmoid function** f, expressed by:

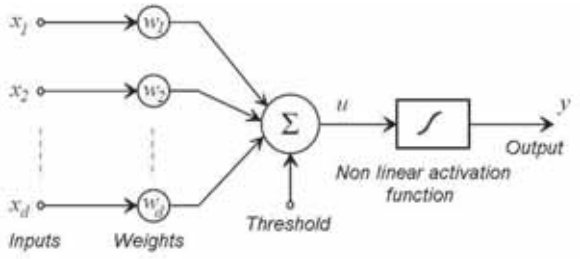


Figure 8.1: Zoom on a perceptron to explain operations of a simple neural network.

$$f(x) = \frac{1}{1 + \exp^{-\beta x}} \tag{8.1}$$

where β is a parameter to choose, meaning the scale value applied by the neuron. β is usually chosen equal to 1.

The **backpropagation** algorithm [128] provides a means of training the network to perform supervised learning tasks. Supervised learning starts with the presentation of a set of example input features to a learning system. The learner's output is then compared with the known correct output for each pattern, and some adjustments are made so as to improve the response of the learner to those patterns. The MLP is trained in order to give the value 1 to the output of the node corresponding to the true class and 0 to the others. In practice, each output has a value between 0 and 1 representing the confidence level that the character belongs to the corresponding class. The architecture of the multi-layer perceptron is displayed in Figure 8.2. In the classifier, we use the improved backpropagation of Vogl et al. [154] to speed up the training step.

In order to recognize many variations of the same character, features need to be robust against noise, distortions, style variation, translation, rotation or shear. Invariants are features which have approximately the same value for samples of the same character, deformed or not. To be as invariant as possible, input-characters are normalized into a $N \times N$ size with $N = 16$. However, not all variations among characters such as noise or degradations can be modelled by invariants, and the database used to train the neural network must have different variations of a same character.

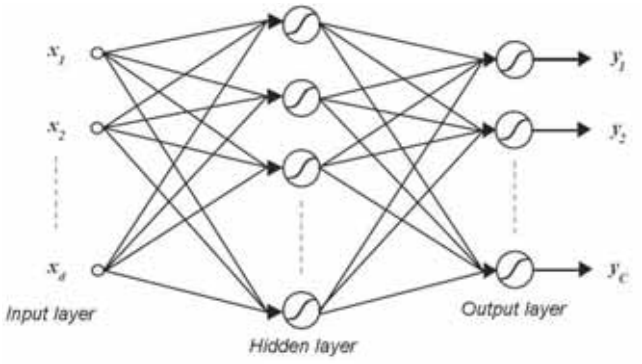


Figure 8.2: Architecture of a multi-layer perceptron where x_i are inputs and y_i are outputs.

In experiments, we use the feature extraction of Gosselin [48] which is based on **contour profiles**. The feature vector is based on the edges of characters and **a probe** is sent in each direction (horizontal, vertical and diagonal) and to get the information of holes like in the 'B' character, some interior probes are sent from the center. Moreover, another feature is added: the ratio between original height and original width in order to very easily discriminate an 'i' from an 'm'. Explanations of probes are given in Figure 8.3.

Experimentally, in order to lead to high recognition rates, we complete this feature set with Tchebychev moments, which are orthogonal moments. Moment functions of a 2D image are used as descriptors of shape. They are invariant with respect to scale, translation and rotation. Traditional orthogonal moments are based on Legendre or Zernike radial polynomials.

According to [103], we use instead Tchebychev moments of order 2 for their robustness to noise, and the general definition for order $p+q$ (p and q varying from 0 to N-1) and for the image $I(i,j)$ is given by:

$$T_{pq} = \frac{1}{\rho(p,N)\rho(q,N)} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} t_p(i) t_q(j) I(i,j) \qquad (8.2)$$

where $\rho(m,N)$ is the square norm of $t_m(x)$, the scaled Tchebychev polynomial and defined by:

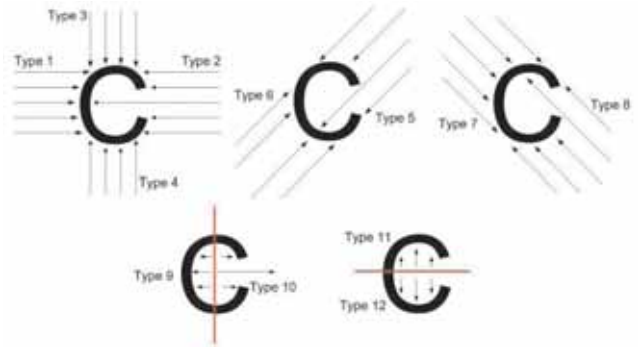


Figure 8.3: The probes characteristics used to extract character contour.

$$\rho(m,N) = \frac{N(1-\frac{1}{N^2})(1-\frac{2^2}{N^2})\dots(1-\frac{m^2}{N^2})}{2m+1} \tag{8.3}$$

and t_m with the recursive relation:

$$(m+1)t_{m+1}(x) - (2m+1)t_1(x)t_m(x) + m(N^2-m^2)t_{m-1}(x) = 0 \tag{8.4}$$

where $t_0(x) = 1$ and $t_1(x) = 2x - N + 1$.

No feature selection is defined and the feature set is a vector of 63 values provided to an MLP with one hidden layer of 120 neurons (stated by trial-and-error) and an output layer of size 36 for each Latin letter and digit. Due to few training samples for capital letters, uppercase and lowercase letters were initially grouped into the same class. Nevertheless, with the algorithm described in the next paragraph, an output layer of 62 neurons may be considered efficiently. The total number of training samples is 40614 divided into 80% for training only and 20% for cross-validation purpose in order to avoid overtraining [48]. Samples of the training database are built from various data sets, different from the databases detailed in Chapter 4, used in experiments.

Zoom on training database: how to build a relevant and general one?

Traditional database increasers are based on geometrical deformations such as affine transformations or on the reproduction of a degradation model such as [141] to mimic low resolution. In NS images, the very large diversity must be handled and character extraction of a huge data set is awkward and difficult to achieve. Hence, we increase the NS database with the image analogies of Hertzmann et al. [51], with the particular algorithm of texture-by-numbers. The image analogies are, by the way, close to example-based super-resolution experienced in *SISO* interpolation, as briefly explained in Chapter 5.

Given a pair of images A and A', with A' being the binarized version of A, the textured image in the algorithm, and B' the black and white image to transfer texture, the texture-by-numbers algorithm applies texture of A into B' to create B. Binary versions are composed of pixels having values of 0 or 1; texture of A corresponding to areas of 0 of A' will be transferred to areas of 0 of B' and similarly for 1. Multiscale representations through Gaussian pyramids are computed for A, A' and B' and at each level, statistics for every pixel in the target pair (B,B') are compared to every pixel in the source pair (A,A') and the best match is found. Additional mathematical information may be found in [51].

One sample used to increase the training database is displayed in Figure 8.4, which also schematises the concept of image analogies.

The entire process of increasing database is firstly based on character extraction from a given data set, using SMC algorithm of Chapter 6. Characters are hence binarized and normalized. Deformations on character thickness, slant, rotation, and perspective are then performed and the texture-by-numbers is applied on each binary image. A huge and new data set is hence built. To provide standardized characters, all newly-textured characters are then binarized always using the SMC algorithm, leading to realistic degradations of NS images, which enables to increase the database as naturally as possible.

Based on the finite steps of variation for each of the pre-cited parameters, for one extracted character and one given texture, 33480 samples may be created. Hence, the power of increasing database of this method is very large (almost infinite depending on

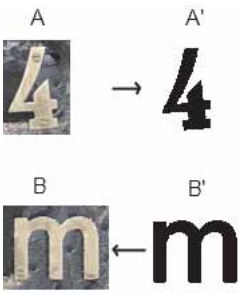


Figure 8.4: Principe of image analogies in the context of database increase. A represents the textured and segmented character, A' its binary version. From a binary version of an 'm' in B', the texture is transferred onto B, similar to the analogy between A and A'.

the parameter variation and the number of textures). Some tests have been done on recognition and rates are slightly increased. Extensive studies are needed to know if the increase is due to the enlarging database and/or the representativeness of the database with texture transfer. Nevertheless, this technique enables the growing of a database in a fast and reliable way.

Zoom on cut characters: how to handle their large number in NS images?

In NS images, acquired by HIDs, the number of cut characters is larger than that in traditional scanner-based acquisition, for several reasons, the main one being a natural cut done by the constrained field of view or by an inaccurate acquisition such as the one taken by a blind user. Some cut characters may also appear from the text detection and localization or text extraction steps. We performed a study in [85] using a prolongation-based approach, where a detection step of cut characters is done at first, followed by prolongation based on curve fitting, and finally a vote between dedicated MLPs for cut characters and the MLP for prolonged characters is performed to help recognition. This technique slightly improves recognition rates. However, prolongations are based on learnt prolongations on typical characters and they fail in case of noisy characters or very artistic fonts or confusion exists between similar characters ('5' and 'S' for example). Hence, we

may conclude that this solution is highly efficient in a dedicated application with the font estimation.

8.1.3 Conclusion on considerations of character recognition

As character recognition was not the central focus of this text, several studies around an efficient MLP-based recognizer have been performed.

Robust OCR is useful for the character segmentation-by-recognition of Chapter 7 and the one described in this chapter is efficient enough to choose the right bandwidth for log-Gabor filters. In NS text recognition, the training database needs to be very representative and an algorithm dedicated to increase data sets in a quasi-natural way has been detailed in this section. Moreover, features need to be robust against degradations and usual affine transformations. Given the complexity of the study for handling cut characters, features are also required to be insensitive to cut characters.

Several improvements may be brought for this step such as the development of a light, efficient and versatile recognizer to use in low computational resources devices. Additional works in feature extraction and selection are also required. An extensive study on the compromise of color-based character recognition and the framework may also be done in terms of efficiency, complexity, required resources and computation time.

Finally, character recognition alone is hardly error-free and linguistic information needs to be added to correct errors for which a light and modular solution is proposed in Section 8.2.

8.2 Recognition-by-Correction

8.2.1 Context of OCR correction

Even a robust OCR is error-prone in a lower percentage and a post-processing correction solution is necessary. Main ways of correcting pattern recognition errors are either multiplication of classifiers to statistically decrease errors by adding information from different computations, such as Lopresti and Zhou proposed for WWW images [81], or by exploiting linguistic information in the special case of character recognition. Depending on applications following text analysis, if too many errors are present, the

result could be completely useless. For example, Lin [77] assessed the impact of imperfect OCR on part-of-speech tagging, essential component of text-to-speech (TTS) systems and he concluded that "the quality of OCR directly affects the performance of the [Natural Language Processing] NLP in the complete content understanding". Hence, minimum error is required for efficient text analysis and speech synthesis.

Commercial OCR use mainly a dictionary to validate in-dictionary words, correct obvious errors (uppercase/lowercase letters, punctuation and so on) and ask the user for feedback on more erroneous words. Processing of all these steps on a standard computer is permitted but on a low-resource device, it is not! For automatic and embedded Chinese sign recognition [165], user feedback is also required through dedicated interfaces for out-of-vocabulary (OOV) words, meaning that no correction is done. The OCRSpell of Taghva and Stofsky [142] builds dynamic confusion by allowing a user to set default statistics for a particular document set. This guarantees that the used statistics will be adequate to find correction for most of the errors in the document.

User intervention may be awkward for industrial purposes and needs to be used in extreme cases when an application requires no error to minimize this expensive intervention. Moreover, for blind people, it makes no sense to require intervention.

There are essentially two types of word errors: non-word errors and real-word errors. A non-word error occurs when a word does not correspond to any valid word in a given word list while a real-word error occurs when a word is interpreted as a valid word in a dictionary, but is not identical to the printed word.

Most papers deal with non-word errors as we will mention in the following subsections and statistical language models are obviously used in real-word errors with part-of-speech tagging into syntactic categories (noun, verbs, ...) and word n-grams. More details for real-word errors will be discussed in Subsection 8.2.4 and may be found in the excellent survey of Kukich [70].

For non-word errors, two main categories are highlighted: isolated word correction and context-dependent word correction. The first one corrects words without any context by taking the one with the highest rank after applying any algorithm while the second one exploits all words in the same sentence, where grammatical context of each leads to meaningful sentences. Similar methods are applied for context-dependent non-word errors and real-word errors.

For isolated word correction, a lexicon is often used and the most simple method is lexicon lookup which gives the existence of a word in the lexicon or not. This point is often the first step of correction as error detection. Generation of candidate corrections and their ranking follow this part. Lightweight methods prefer not to use a lexicon even if dictionary-based methods have low error rates, because they suffer from large storage demands and high computational complexity. Probabilistic techniques and n-gram analysis are hence exploited, among the five kinds of isolated word correction. They use transition and confusion probabilities using hidden Markov models (HMM) (1) or the well-known Viterbi algorithm (VA)(2), first used by Neuhoff [107] in text correction. Thillou et al. [143] have developed a without-lexicon correction based on letter trigram analysis through VA and by exploiting 3-best OCR outputs to add information for correction. A priori probabilities were computed using a French database from a 10-year news archive of "Le Monde" and a Katz smoothing for absent trigrams. The main trouble these methods pose is the absence of a lexicon, and real words may be corrected into OOV words, because some trigrams of the word are less probable, and the incapacity of correcting words with several errors is also a challenging issue. Errors are effectively propagating along the word and even the sentence, if space is considered as a letter.

The three other categories are minimum edit distance techniques(3), enabling the handling of the insertion, deletion or substitution of letters, with the Levenshtein distance [73] for example, the rule-based techniques (4), as properly named, exploiting rules, confusion list and finally neural networks techniques (5), which learn the confusion, directly from rule-based methods.

The last three categories have the main advantage, along with the presence of a lexicon, to handle confusion lists and usual OCR errors such as insertion, deletion and substitution of letters. The challenge is therefore to decrease error rates using a lexicon but in a light way to efficiently and quickly work in HIDs.

Methods similar to ours may be mentioned to highlight advantages of the proposed solution in Subsection 8.2.2. Bunke [11] built an automaton (a particular finite state machine) to find out the most similar strings from a vocabulary using minimum edit distance techniques. An efficient conventional parsing method was allowed in his algorithm, yielding the required results. Nevertheless, a high spatial cost was mentioned as the correction was not driven by errors but by similarity only. Jones et al. [58] described

an OCR post-processing system which uses individual steps for correction: rewrite rules, correction of word split errors and use of word bigram probabilities. The three phases interact with others to guide the search but decision has to be taken at each step.

More similar to the system, by considering an end-to-end generative model, is the one of Kolak et al. [68]. They use at run-time a single transducer that takes a sequence of OCR characters as input, and returns a lattice of all possible sequences of real words as output, along with their weights. This transducer is the result of the off-line composition of several transducers trained separately on the same corpus. The main idea of this system is to split each in-dictionary word into its two most probable subsequences of characters (*e.g.*, "example" ⇒ "ex | ample" and "exam | ple"), and to propose, from the training corpus, a list of observed corrupted sequences (*e.g.*, "exam" ⇒ "exain", "cxam", etc.). A first drawback of this system is perhaps this OCR confusion model, which is greatly context-dependent. The second drawback is surely the off-line composition of a single transducer, because no simplification between the different steps is still possible at run-time.

All aforementioned methods consider OCR as a "black box" and start correction independently of OCR results, missing direct information. We shall propose an efficient and lightweight non-word error correction combining OCR outputs and linguistic information using a lexicon. For this purpose finite state machines are exploited and extensions for context-dependent non-word and real-word error correction will also be finally discussed in Subsection 8.2.2.

8.2.2 Lexicon-based non-word error correction

A finite state machine (FSM) contains a finite number of states and produces outputs on state transitions based on inputs. FSMs are widely used to model systems in diverse areas, such as text-to-speech, text processing, communication protocols or data compression (a lexicon is represented by one machine only). They often lead to a compact representation of rules, which can be lexical for example, which is considered as natural by linguists.

FSMs are one of the most widely used models in computer programming in general; they have even been adopted as a part of the well-known Unified Modelling Language (UML).

FSMs include finite state automata (FSAs) and finite state transducers (FSTs) with their weighted components, respectively

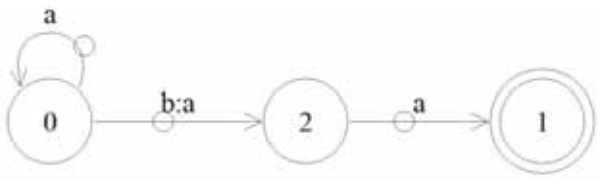


Figure 8.5: Display of a simple FST: this transducer contains 3 states with an alphabet of 2 letters (a, b). b:a means b may be transduced into a.

WFSAs and WFSTs.

FSMs are defined by a 7-tuple (Q, i, F, Σ, Δ, δ, σ):

- Q is the set of states.
- $i \in Q$ is the initial state.
- $F \subseteq Q$ is the set of final states.
- Σ and Δ, finite sets corresponding respectively to the input and output alphabets of the machine, with alphabet, meaning a finite set of symbols. Elements of an alphabet are called words and a subset of an alphabet is called a language. In an automaton, Δ is undefined.
- δ is the state transition function which maps $Q \times \Sigma$ to Q.
- σ is the output function which maps $Q \times \Sigma$ to Δ^* (meaning Δ and the empty word, ε).

Figure 8.5 illustrates[1] a simple FST with $Q = \{0, 1, 2\}$, $i = \{0\}$, $F = \{1\}$, $\Sigma = \Delta = \{a, b\}$, $\delta(0, a) = 0$, $\delta(0, a : b) = 2$, $\delta(2, a) = 1$, $\sigma(Q, \Sigma) = a^*ba|a^*$, meaning that all output words are composed of an infinite number of a and may be followed by ba or not. This basic example shows the modelling possibility of OCR errors.

WFSMs are finite state machines in which each transition has a weight and they are particularly suited to integrate probabilities between transitions. Without weights, all generated word sequences in NS text correction will be equivalent if several ones

[1]Figures of FSMs are produced by the Dotty software, a graph layout product, which may be found at http://hoagland.org/Dot.html.

belong to the lexicon. By including weights, traducing a priori probabilities, the best path with the highest probability could easily be chosen, similarly as VA which looks for the best path, the one with the minimum weight.

To combine or decode several FSMs, the composition operation (o) defined for mappings is very useful. It allows constructing more complex FSMs from simpler ones, hence becoming a complex FST. The result of the application $\tau_1 o \tau_2$ to a string can be computed by first considering all output strings associated with the inputs in τ_1, then applying τ_2 to all these strings. It allows an end-to-end process by considering all generated possibilities.

Other operations, such as union and equivalence, are out of scope of this text and may be found in [101, 102] along with detailed explanations on FSM in general.

For recognition-by-correction, the proposition is based on combination of 4 FSMs:

1. The first one is a dynamic WFSA α_1 which links 3-best OCR outputs by combining all word possibilities. This FSA is dynamic as the OCR output is not known a priori for a given character and weighted as dedicated weights for each output are given. The best output has to be obviously privileged against the second and third ones. Hence, we experimentally award a weight 3 times larger for the best output than that given to the second and third outputs. Finer probabilities may be defined based on a large corpus. Weights of second and third outputs are slightly different as OCR robustness does not enable the awarding of very different weights.

   Ex: For the word 'late', best outputs were 'lelo', second ones 'tate' and third ones 'hoha' and the dynamic WFSA may be represented as in Figure 8.6:

   where $Q = \{0, 1, 2, 3, 4\}$, $i = \{0\}$, $F = \{1\}$ and $\Sigma = ASCII$.

2. The second one τ_2, a static WFST, represents a static confusion list with predefined rules based on classical confusions such as 'i' and 'l', 'rn' and 'm' and so on. Weights are defined to differentiate very probable confusions and less probable ones and have been computed on DB-ICDAR and based on the home-made OCR errors.

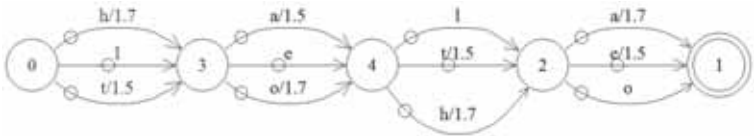


Figure 8.6: Display of the first weighted automaton of the system to model the 3-bests OCR output for the example 'lelo'.

3. The third machine τ_3 is a static WFST to reaccent characters. In NS text understanding with all possible noise in an image, accents are difficult to extract and frequent errors of absent accents occur. Moreover, in advertisements for example, some words may be all in capital letters without accents. This machine is useful for language with accents such as French. Reaccentuation is weighted to privilege in-dictionary words without accents.

4. Finally the fourth machine τ_4 represents a lexicon (with accented words) by an FST of 330 278 lexical forms, including all syntactic classes, singular and plural and all inflected forms of verbs.

The best output is straightforward: all machines are composed together with $\alpha_1 o \tau_2 o \tau_3 o \tau_4$ and results are all possible in-dictionary accented words from OCR outputs. The best one with the minimum weight will be chosen by multiplying consecutive weights (in practice, weights are given by a negative logarithm and they are simply summed).

8.2.3 Evaluation

In order to assess recognition-by-correction using the 3-best OCR outputs as inputs in the lexicon-based non-word error correction, we detail some examples from DB-WWW and DB-Sypole with French text[2]. Comparisons are done with recognition without correction using the home-based recognizer, followed by the use of a correction without lexicon, as the one described in [143] which uses trigram analysis through VA, and with the proposed correction technique in this section. Table 8.1 displays results. Five

[2]Only a French dictionary was available.

expressions have been carefully chosen to illustrate the complexity of correction and modularity and efficiency of FSM.

The first one is a sentence with proper nouns, hence OOV words. The result of OCR without correction is correct. With VA-based correction, the first name is changed to another one, which is wrong. The same bad result appears with FSM-based correction, which forces the correction to reach a real word. Nevertheless, integration of an additional machine with syntactic information may resolve this issue. The proper noun 'mannoni' is converted to 'marnons', meaning the inflected verb of 'marner' ('we marl'). Hence, with a syntactic correction, the first name could not be followed by this inflected form. For the second sentence, correction may be found either in the 3-best output of OCR or in the confusion list. FSM-based correction gives better results as the word 'proqrammation' does not exist in a lexicon. The third example has its correction only in the confusion list as the error was due to the presence of a cut character on image edges while the fourth one only in the 3-best OCR output. Finally, the fifth expression is never corrected as each word is a real-word. Only with an expression-based lexicon, this correction may be handled: the right expression is "coup de foudre" meaning "love at first sight", which is the second best solution of FSM-based correction.

An interesting study has been done on correction constraints. Some strongly erroneous words, with the confusion list stated for τ_2, may be turned into real words. Nevertheless, these resulting words are very far from the initial word. The same consequence may appear for OOV words. Hence, in order to automatically prune the composed machine, we restrain the number of corrections at around one third of the word length. The machine is hence heavier but the number of explored paths drastically decreases. The initial complexity of the solution is dependent on the number of letters n and was $O(2^n)$ and is now $O(n^c/c!)$ with c, the number of permitted corrections. Based on several sentences, results were identical but with constrained correction, they were computed much faster! If results are not in-dictionary words, other decisions may be taken, in order not to obtain very different words. Is it an OOV word? Is it an incorrect separation between words? If the last answer to the last question is positive, another machine, modelling space insertion and deletion, may then be added. In the study, if corrected words are very far from the initial recognized ones, we decide not to apply correction. If the end application is speech synthesis, the spoken word may still be understood by

Table 8.1: Comparison of correction samples based on several types of correction.

	Sentences	♯ errors
Without correction	"octave mannoni"	0
	"les 100o fonctlons de proqrammatlon"	4
	"tout sur la programmatior"	1
	"un feu iouge"	1
	"loup de foudre"	1
With VA-based correction [143]	"ociave mannoni"	1
	"les 1000 fonctions de proqrammation"	1
	"tout sur la programmatior"	1
	"un feu louse"	2
	"loup de foudre"	1
The FSM-based correction	"octave marnons"	2
	"les 1000 fonctions de programmation"	0
	"tout sur la programmation"	0
	"un feu rouge"	0
	"loup de foudre"	1

Table 8.2: Comparison of correction samples based on the number of included recognition outputs (either only one or three).

	Sentences	♯ errors
With 1-best OCR only output	"octave marnons"	2
	"les 1000 fonctions de programmation"	0
	"tout sur la programmation"	0
	"un feu fouge"	1
	"loup de foudre"	1
The recognition-by-correction	"octave marnons"	2
	"les 1000 fonctions de programmation"	0
	"tout sur la programmation"	0
	"un feu rouge"	0
	"loup de foudre"	1

users based on the initial word. A particular threshold has been chosen to balance the compromise between the number of corrected errors, which may be high, leading to very different words and the pronunciation of the recognized word, which may be close to the real one.

Since the system does not consider OCR as a black box, a comparison of results with the system and a composition of FSMs not using this information is done. Actually, the first WFSA α_1 is constrained to only 1-best OCR output. The results, shown in Table 8.2, highlight the relevance of this additional machine, compared to recent efficient and similar existing techniques. Moreover, without this additional information, on several other examples, results were empty meaning that no word was found in the lexicon.

To appreciate the efficiency of an FSM-based correction, we computed recognition rates for French parts of DB-WWW and DB-Sypole. Results are displayed in Table 8.3 and the improvement compared to a non-dictionary based correction, such as the VA one, is clearly proven. Main serious errors of VA-based correction were to correct real words into non-real words and its incapacity of modelling N towards M correction, meaning that insertion and deletion may not be handled with only the confusion list inside VA. Errors in FSM-based correction are mainly due to real OOV words, either due to cut words or foreign words

Table 8.3: Recognition rates (%) without correction, with the VA-based one and with the FSM-based one.

	No correc.	VA-based correc.	FSM-based correc.
Rec.	83.4%	86.5 %	94.7 %

and acronyms.

Finally the main purpose was computation time and low resources to get a good compromise between high quality correction using a lexicon and reasonable computation to fit HID resources.

Much effort has been provided by the Multitel TTS team[3] to build an efficient, lightweight and fast toolkit. On a classical PDA (on PocketPC2003, 520 MHz), the algorithm corrects a word of 13 letters in 0.2s after loading the composed machine. Finer details of computation time in different applications for this FSM toolkit may be found in [7, 8].

8.2.4 Conclusion on recognition-by-correction

The lexicon-based non-word error correction is very modular, which enables to handle OCR errors, such as insertion, deletion or substitution of letters. Moreover, the system exploits 3-best OCR outputs to be directly linked with recognition, which is a real innovation and includes weights by regarding the output rank. It enables to correct more words with highest probability. Regarding recognition-by-correction, detailed in Subsection 8.2.2, more thorough work is required to very accurately tune each machine. Nevertheless, results are drastically improved and the modularity offered by FSMs is very promising for NS text correction.

Several extensions may be formulated: similar to Kolak et al. [68], higher-level syntactic information may be built through additional FSMs to handle word segmentation problems for example, or to choose the best output regarding the whole sentence as well. Syntactic information, modelled by an additional machine, with grammatical forms, could be also used to correct real-word errors! To be even more efficient and fast, this step may be included in the syntactic analyzer, present in all TTS techniques, if applications of NS text understanding end into a TTS algorithm.

[3] http://www.multitel.be/TTS/

8.3 Conclusion

In the first part of this chapter, we detailed the proposition of algorithms of NS character recognition with relevant study on cut characters handling and solutions to increase NS databases. Moreover, in the second part, we presented a recognition-by-correction method using modular and lightweight FSMs, carefully highlighted with an evaluation part resulting with an increase in recognition rates. The main points to highlight are the following:

- Brief explanation of the in-house OCR, leading to satisfying results in terms of bandwidth determination for log-Gabor filters.
- Recognition part was not described in terms of competitive recognition performance but in terms of inclusion inside an NS text understanding. Thorough studies have been made to increase recognition rates:
  - Artificial but realistic increase of database samples based on image analogies technique to synthesize degradations of natural scene images.
  - Handling of cut characters through parallel neural networks. Results are slightly improved but confusion between similar patterns remains an essential issue to complete cut characters.
- Correction of recognition errors is an essential part of pattern recognition to decrease the number of errors and to add robustness through different information. The FSM-based correction arrives at the following points:
  - Recognition rates increase
  - Lightweight use of a lexicon
  - Correction of insertion, deletion and substitution errors
  - Exploitation of 3-best OCR outputs to combine information and to statistically converge towards best corrections.
- The final recognition-by-correction algorithm could also be exploited in conventional OCR, not specifically for NS text recognition.

Finally, these two steps finish off the whole process of NS text understanding from resolution enhancement to OCR correction.

— CHAPTER 9 —

Conclusion

This last chapter aims at concluding this text by summing up each contribution among smaller conclusions for each step. The first part highlights important points according to us to realize an efficient and versatile NS text understanding and the second parts emphasizes interesting work prolongations in other image processing fields and the focus to give in next years.

9.1 Conclusions and Contributions

Color variation sources have been detailed by considering the triplet Light, Object and Camera in order to understand image formation. Moreover, based on MacAdam ellipses experiments, requirements for an efficient text understanding came out:

- Handling of NS degradations in a combined way is preferred, instead of independently in order to decrease the number of errors and decisions to take at each step.
- Solution using chromaticities and luminance information without defining a new color space is expected to combine these complementary sources and to circumvent awkward definitions of color spaces.
- Exploitation of magnitude and relative orientation of colors to handle color variations in a more efficient way could be a novel solution.

After browsing main methods in text binarization and more particularly in NS text extraction in another part, we concluded that some points were missing to properly handle NS text images:

- Handling of all degradations with a computationally interesting grouping-based methods is expected.
- Recent papers dealing with spatial information enable obvious correction of bad extraction and spatial information among chromaticities and luminance one is mandatory.
- Consideration of very complex and low resolution images instead of simple or middle-difficulty NS images is a hole in NS text understanding evaluation.
- Decrease of the number of rules in algorithms, which is contrary to versatility, especially in character segmentation techniques is mandatory as text properties are not efficiently exploited.

A deep study on impact of color spaces on NS text extraction highlighted the inappropriate definition of color spaces. Recent color spaces, even perceptual ones, handle better more complex images but not in a general way and sometimes perform very poorly on simple images.

Based on these considerations we proposed solutions for color variations, complex backgrounds, low resolution and very joined characters, which are main failures reasons of text understanding. Hence, we presented the selective metric clustering algorithm which exploits magnitude of color pixels along with their orientation to introduce hue values inside the RGB color space. Color is hence fully used for text extraction from background. In order to add finer accuracy to separate characters into individual components, color is associated with intensity and spatial information in the subsequent step of character segmentation. A particular effort has been done to intermingle each step to increase overall robustness and these links have proven their efficiency through detailed results in each chapter. For evaluation, databases have been carefully chosen in order to highlight versatility. A large public database was used among smaller ones such as samples from Internet and camera-based images from low-resolution cameras. Independence against a particular database is hence ensured.

Text understanding and more particularly natural scene text extraction need pre- and post-processing steps to correct or circumvent some degradations and to add useful information in order to increase final recognition rates. Hence, from Chapter 5 to Chapter 8, we dealt with available information in order to improve global quality of text extraction:

Resolution Enhancement: By applying simple affine motion assumption instead of pure translational model between video frames, we presented the SURETEXT algorithm which enhances high frequency information inside low resolution video. Super-resolution with Teager filtering has proven its efficiency against traditional still images resolution increaser and classical super-resolution algorithms. The method is quite lightweight compared to more complex and recent algorithms which assume information from camera (point-spread function for example) and Bayesian reconstruction methods. Nevertheless, more efforts have to be done to decrease computation time even more in order to enable inclusion of super-resolution algorithms into current and future handheld imaging devices.

Text Extraction: This step was the main focus of the work as its quality immediately has an impact on recognition results, meaning that recognition rates may be satisfying only if text extraction presents very good results. Through the parallel use of three extraction hypotheses issued from an Euclidean-based and angle-similarity-based clustering and a global thresholding, we introduced the SMC algorithm and showed its performance with detailed results in each subsection. Several color spaces into different clustering algorithms have been tested to highlight the sufficiency of RGB for handling a very large set of natural scene images. Similarly, several clustering metrics have been put to the test through several quality measures to point out which metrics were the best ones to answer desired versatility. To be completely included into a consistent text understanding, the main goal is now to meet text detection and extraction results with Potts model for example. Text detection is rarely based on color information alone and texture and edges are often exploited, which perfectly complements studies in this text.

Figure 9.1: Sample of screen-rendered text where first promising results have been computed by using the log-Gabor-based character segmentation.

Unit-based Segmentation: Even if this step is missing in most algorithms, results have been shown to highlight the usefulness of character segmentation inside very complex natural scene images. In literature, complex samples of the public database we used have never been presented and by using log-Gabor filters and intimately linked to the text extraction step, we introduced promising results for these images. First encouraging results even appeared after requests during a conference to analyze screen-rendered text such as the one displayed in Figure 9.1. To fully exploit results of this step, one future work is to use log-Gabor features, dynamically tuned by our method, as feature classifier inside a recognition framework.

NS Character Recognition and Correction: This final step has been studied through several works to add neighboring information in order to increase recognition rates. Natural scene images are often populated with cut characters and we stated that correction of cut characters may be a solution but robust recognition for cut characters is preferred. A particular attention has been paid to build a large and realistic training database for a supervised classification (with multi-layer perceptron for example). Correction of recognition outputs is a necessary step to go towards efficient recognition for text-to-speech algorithms, for example. By adding linguistic information through lightweight finite state machines, we showed a drastic improvement of recognition results (from 83.4% to 94.7%) and highlighted the modularity of this solution.

9.2 Interesting Prolongations and Discussion

Among future works of each step detailed in the previous section, one of the main prolongation work will be to extend some of these solutions for extraction of other objects in natural scene images to show once again versatility of these methods. Obviously, character segmentation or text correction are dedicated steps of text analysis. Nevertheless, the combination of color, intensity and spatial information or handling of low resolution frames may lead to interesting results for other applications.

In order to sum up required points to extend to build an efficient NS text understanding, we may highlight that low-resolution still images (due to the low resolution acquisition or a small detail in a high resolution image) require additional work. About the global system and if resources are available, the small amount of errors at each step may be decreased by keeping information until recognition errors correction. These additional hypotheses will be handled through another step of information fusion, such as simply the solution with the least correction to reach real words.

Due to the great expansion of electronic goods and their ever increasing performance, readers may wonder if these text topics will not be obsolete in a few years. In some recently launched smartphones in Asia with 3.2 Megapixels cameras and rudimentary embedded OCR or with expansion to 8 Megapixels of consumer-grade digital cameras, is the resolution enhancement step useful? For tiny characters or correction of degradations, the merge of several frames is still interesting. The text extraction part handling complex backgrounds and uneven lighting will be necessary for a long time: professional expensive cameras have still problems with illumination by nature and complex backgrounds, especially in advertisements. Such issues will not disappear anytime! Unit-based segmentation may be removed by other computationally very demanding methods but character recognition and correction is mandatory to understand text. Hopefully, text understanding steps will be automatically embedded into handheld imaging devices soon for exciting and useful applications in daily life!

BIBLIOGRAPHY

[1] *Robust reading competition*, 2003. `http://algoval.essex.ac.uk/icdar`.

[2] A. Abadpour and S. Kasaei. A new parametric linear adaptive color space and its implementation. In *Proc. Annual Computer Society of Iran Computer Conf.*, pages 125–132, 2004.

[3] S. Antani, D. Crandall, and R. Kasturi. Robust extraction of text in video. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 831–834, 2000.

[4] A. Apodaca and L. Gritz. *Advanced RenderMan: Creating CGI for Motion Picture*. Morgan Kaufmann Publishers Inc., 1999.

[5] J.H. Bae, K.K. Jung, J.W. Kim, and H.J. Kim. Segmentation of touching characters using an MLP. *Pattern Recognition Letters*, 19(8):701–709, 1998.

[6] S. Baker and T. Kanade. *Super-resolution optical flow*, 1999. Tech. Report CMU-RI-TR-99-36.

[7] R. Beaufort. *Compilation de Règles de Réécriture en transducteurs à états finis*, 2006. Multitel tech. report.

[8] R. Beaufort. *FSM Library: description de l'API*, 2006. Multitel tech. report.

[9] P. Berkhin. *Survey of clustering data mining techniques*, 2002. Tech. report, Accrue Software.

[10] L. Bottou, P.Haffner, Y. Le Cun, P. Howard, and P. Vincent. Djvu: An image compression system for distributing scanned document on the internet. In *Proc. Actes de la Conf. Int. Francophone sur l'Ecrit et le Document*, 2000.

[11] H. Bunke. Fast approximate matching of words against a dictionary. *Computing*, 55(1):75–89, 1995.

[12] D. Capel and A. Zisserman. Super-resolution enhancement of text image sequences. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 600–605, 2000.

[13] D.P. Capel. *Image mosaicing and super-resolution*. PhD thesis, University of Oxford, 2001.

[14] R.G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(7):690–706, 1996.

[15] T.F. Chan and C.K. Wong. Total variation blind deconvolution. *IEEE Trans. Image Processing*, 7(3):370–375, 1998.

[16] D. Chen. *Text detection and recognition in images and video sequences*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2003.

[17] M-C. Chiang and T.E. Boult. Local blur estimation and super-resolution. In *Proc. IEEE Computer Vision and Pattern Recognition*, pages 821–826, 1997.

[18] C.-L. Chin and C.-T. Lin. Detection and compensation algorithm for backlight images with fuzzy logic and adaptive compensation curve. *Int. Jour. Pattern Recognition and Artificial Intelligence*, 19(8):1041–1057, 2005.

[19] D. Comaniciu. *Nonparametric robust methods for computer vision*. PhD thesis, Rutgers University, 2000.

[20] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *Proc. Int. Conf. on Machine Learning*, 2006.

[21] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Jour. Royal Statistical Society*, 39:1–38, 1977.

[22] E. Diday, J. Lemaire, J. Pouget, and F. Testu. *Eléments d'analyse de données*. Dunod, 1982.

[23] K. Donaldson and G.K. Myers. Bayesian super-resolution of text in video with a text-specific bimodal prior. *Int. Jour. Document Analysis and Recognition*, 7(2–3):159–167, 2005.

[24] F. Drira and H. Emptoz. A recursive approach for bleed-through removal. In *Proc. Camera-based Document Analysis and Recognition*, pages 119–126, 2005.

[25] M. Droettboom. Correcting broken characters in the recognition of historical printed documents. In *Joint Conf. on Digital Libraries*, 2003.

[26] Y. Du. *Text Detection and Restoration of Color Video Images*. PhD thesis, University of Maryland, 2003.

[27] Y. Du, C-I. Chang, and P.D. Thouin. Unsupervised approach to color video thresholding. *Optical Engineering*, 43(2):282–289, 2004.

[28] M. Elad and A. Feuer. Restoration of a single superresolution image from several blurred, noisy, and undersampled measured images. *IEEE Trans. Image Processing*, (12):1646–1658, 1997.

[29] C. Elkan. Using the triangle inequality to accelerate k-means. In *Proc. Int. Conf. on Machine Learning*, 2003.

[30] N. Esaki, M. Bulacu, and L. Shomaker. Text detection from natural scene images: towards a system for visually impaired persons. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 683–686, 2004.

[31] C. Fang. *Deciphering algorithms for degraded document recognition*. PhD thesis, State University of New York, 1997.

[32] S. Farsiu, M.D. Robinson, M. Elad, and P. Milanfar. Advances and challenges in super-resolution. *Int. Jour. of Imaging Systems and Technology*, 14(2):47–57, 2004.

[33] S. Farsiu, M.D. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super-resolutions. *IEEE Trans. Image Processing*, 13(10):1327–1344, 2004.

[34] S. Ferreira, C. Thillou, and B. Gosselin. From picture to speech: an innovative application for embedded environment. In *Proc. ProRISC workshop on Circuits, Systems and Signal Processing*, 2003.

[35] D.J. Field. Relations between the statistics of natural images and the response properties of cortical cells. *Jour. Opt. Soc. Amer. A*, 4(12):2379–2394, 1987.

[36] M. Figueiredo and A.K. Jain. Unsupervised learning of finite mixture models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24:381–396, 2002.

[37] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting applications to image analysis and automated cartography. *Comm. of ACM*, 24(6):381–393, 1981.

[38] L. Fletcher and A. Zelinsky. Super-resolving signs for classification. In *Proc. Australasian Conf. Robotics and Automation*, 2004.

[39] L. Fu, W. Wang, and Y. Zhan. A robust text segmentation approach in complex background based on multiple constraints. In *Proc. Pacific Rim Conf. on Multimedia*, pages 594–605, 2005.

[40] H. Fujisawa, Y. Nakano, and K. Kurino. Segmentation methods for character recognition: from segmentation to document structure analysis. In *Proc. the IEEE*, pages 1079–1091, 1992.

[41] J. Gao, J. Yang, Y. Zhang, and A. Waibel. *Text Detection and Translation from Natural Scenes*, 2001. Techn. Report CMUCS -01-139.

[42] C. Garcia and X. Apostolidis. Text detection and segmentation in complex color images. In *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pages 2326–2330, 2000.

[43] B. Gatos, I. Pratikakis, and S.J. Perantonis. Towards text recognition in natural scene images. In *Proc. Int. Conf. Automation and Technology*, pages 354–359, 2005.

[44] V. Gaudissart, S. Ferreira, C. Thillou, and B. Gosselin. Mobile reading assistant for blind people. In *Proc. Speech and Computer Conf.*, pages 538–544, 2004.

[45] V. Gaudissart, S. Ferreira, C. Thillou, and B. Gosselin. Sypole: a mobile assistant for the blind. In *Proc. European Signal Processing Conf.*, 2005.

[46] T. Gevers. *Color in image databases*, 2000. Isis tech. report.

[47] J. Gllavata, R. Ewerth, and B. Freisleben. Finding text in images via local thresholding. In *Proc. IEEE Symposium on Signal Processing and Information Technology*, pages 539–542, 2003.

[48] B. Gosselin. *Application de réseaux de neurones artificiels à la reconnaissance automatique de caractères manuscrits.* PhD thesis, Faculté Polytechnique de Mons, 1996.

[49] H. Hamza, E. Smigiel, and A. Belaid. Neural based binarization techniques. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 317–321, 2005.

[50] C.J. Harris and M. Stephens. A combined corner and edge detector. In *Proc. Alvey Vision Conf.*, pages 147–151, 1988.

[51] A. Hertzmann, C.E. Jacobs, N. Oliver, B. Curless, and D.H. Salesin. Image analogies. In *Proc. ACM SIGGRAPH, Int. Conf. On Computer Graphics and Interactive Techniques*, 2001.

[52] M. Hild. Color similarity measures for efficient color classification. *Jour. of Imaging Science and Technology*, 15(6):529–547, 2004.

[53] P.V.C Hough. Machine analysis of bubble chamber pictures. In *Proc. Int. Conf. on High Energy Accelerators and Instrumentation*, pages 554–556, 1959.

[54] Mathworks Inc. *Matlab: The language of technical computing*, 1984-. http://www.mathworks.com.

[55] M. Irani and S. Peleg. Improving resolution by image registration. *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, 53(3):231–239, 1991.

[56] K. Iwatsuka, K. Yamamoto, and K. Kato. Development of a guide dog system for the blind people with character recognition ability. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 453–456, 2004.

[57] A.K. Jain, R.P.W Duin, and J. Mao. Statistical pattern recognition: a review. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[58] M.A. Jones, G.A. Story, and B.W. Ballard. Integrating multiple knowledge sources in a bayesian OCR post-processor. In *Proc. Int. Conf. Document Analysis and Recognition*, volume 1, pages 925–933, 1991.

[59] K. Jung, K.I. Kim, and A.K. Jain. Text information extraction in images and video: a survey. *Pattern Recognition*, 37(5):977–997, 2004.

[60] M. Kamel and A. Zhao. Extraction of binary character/graphics images from grayscale document images. *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, 55(3):203–217, 1993.

[61] D. Karatzas and A. Antonacopoulos. Text extraction from web images based on a split-and-merge segmentation method using colour perception. In *Proc. IEEE Int. Conf. Pattern Recognition*, volume 2, pages 634–637, 2004.

[62] D. Keren, S. Peleg, and R. Brada. Image sequence enhancement using sub-pixel displacements. In *Proc. Computer Vision and Pattern Recognition*, pages 742–746, 1988.

[63] H.Y. Kim. Segmentation-free printed character recognition by relaxed nearest neighbor learning of windowed operator. In *Proc. Simposio Brasileiro de Computaçao Grafica e Processamento de Imagens*, 1999.

[64] I.-J. Kim. Multi-window binarization of camera image for document recognition. In *Proc. Int. Workshop on Frontiers in Handwriting Recognition*, pages 323–327, 2004.

[65] I.-J. Kim. *Keynote presentation of Camera-based Document Analysis and Recognition*, 2005. `http://www.m.cs.osakafu-u.ac.jp/cbdar`.

[66] J. Kim, S. Park, and S. Kim. Text locating from natural scene images using image intensities. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 655–659, 2005.

[67] K.K. Kim, J.Y. Lee, and J.H. Kim. Character segmentation of camera document. In *Proc. Signal Processing, Pattern Recognition, and Applications*, 2003.

[68] O. Kolak, W. Byrne, and P. Resnik. A generative probabilistic OCR model for NLP applications. In *Proc. Conf.*

of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, volume 1, pages 55–62, 2003.

[69] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from: `http://www.csse.uwa.edu.au/~pk/research/matlabfns/`, 2006.

[70] K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992.

[71] Y. Kusachi, A. Suzuki, N. Ito, and K. Arakawa. Kanji recognition in scene images without detection of text fields - robust against variation of viewpoint, contrast, and background texture -. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 457–460, 2004.

[72] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. the IEEE*, 86(11):2278–2324, 1998.

[73] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[74] H. Li and D. Doermann. Text enhancement in digital video using multiple frame integration. In *Proc. ACM Int. Conf. on Multimedia*, pages 19–22, 1999.

[75] J. Liang, D. Doermann, and H. Li. Camera-based analysis of text and documents: a survey. *Int. Jour. Document Analysis and Recognition*, 7(2–3):84–104, 2005.

[76] R. Lienhart and A. Wernicke. Localizing and segmenting text in images, videos and web pages. *IEEE Trans. Circuits and Systems for Video Technology*, 12(4):256–268, 2002.

[77] X. Lin. Impact of imperfect OCR on part-of-speech tagging. In *Proc. Int. Conf. Document Analysis and Recognition*, volume 1, pages 284–288, 2003.

[78] B. Lindbloom. Bruce lindbloom website. Available from: `http://www.brucelindbloom.com/index.html`, 2006.

[79] Y. Liu, S. Goto, and T. Ikenaga. A robust algorithm for text detection in color images. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 399–405, 2005.

[80] P.K. Loo and C.L. Tan. Adaptive region growing color segmentation for text using irregular pyramid. In *Proc. Int. Workshop Document Analysis Systems*, 2004.

[81] D. Lopresti and J. Zhou. Using consensus sequence voting to correct OCR errors. *Computer Vision and Image Understanding*, 67(1):39–47, 1997.

[82] D. Lopresti and J. Zhou. Locating and recognizing text in WWW images. *Information Retrieval*, 2:177–206, 2000.

[83] S.M. Lucas and C.R. Jaimez Gonzalez. Web-based deployment of text locating algorithms. In *Proc. Camera-based Document Analysis and Recognition*, pages 101–107, 2005.

[84] S.M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto, H. Miyao, Y. Zu, W. Ou, C. Wolf, J.-M. Jolion, L. Todoran, M. Worring, and X. Lin. Icdar 2003 robust reading competitions: entries, results and future directions. *Int. Jour. Document Analysis and Recognition*, 7(2–3):105–122, 2005.

[85] A. Luijkx, C. Thillou, and B. Gosselin. A prolongation-based approach for recognizing cut characters. In *Proc. Int. Conf. on Computer Vision and Graphics*, 2004.

[86] R. Lukac, B. Smolka, K. Martin, K.N. Plataniotis, and A.N. Venetsanopoulos. Vector filtering for color imaging. *IEEE Signal Processing, Special Issue on Color Image Processing*, 22(1):74–86, 2005.

[87] X.-P. Luo, J. Li, and L.-X. Zhen. Design and implementation of a card reader based on build-in camera. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 417–420, 2004.

[88] C. Mancas-Thillou and B. Gosselin. Color text extraction from camera-based images - the impact of the choice of the clustering distance -. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 312–316, 2005.

[89] C. Mancas-Thillou and B. Gosselin. Character segmentation-by-recognition using log-Gabor filters. In *Proc. IAPR Int. Conf. Pattern Recognition*, 2006.

[90] C. Mancas-Thillou and B. Gosselin. Spatial and color spaces combination for natural scene text extraction. In *Proc. Int. Conf. Image Processing*, 2006.

[91] C. Mancas-Thillou and B. Gosselin. Color text extraction with selective metric-based clustering. *Computer Vision and Image Understanding, Special issue on Color Image Processing*, 2007. To appear in February 2007.

[92] C. Mancas-Thillou, M. Mancas, and B. Gosselin. Camera-based degraded character segmentation into individual components. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 755–759, 2005.

[93] C. Mancas-Thillou, M. Mancas, and B. Gosselin. Segmentation en caractères individuels dans des images de scènes naturelles. In *Proc. Colloque sur le Traitement du Signal et des Images*, 2005.

[94] C. Mancas-Thillou and M. Mirmehdi. Super-resolution text using the Teager filter. In *Proc. Camera-based Document Analysis and Recognition*, pages 10–16, 2005.

[95] C. Mancas-Thillou and M. Mirmehdi. *An introduction to super-resolution text in Recent Advances in Digital Document Processing*. Springer-Verlag, 2006. To appear, in press.

[96] J.C. Maxwell. On the theory of three primary colors. *Proc. Roy Inst.*, 3:370–375, 1858–1862.

[97] S. Messelodi and C.M. Modena. Automatic identification and skew estimation of text lines in real scene images. *Pattern Recognition*, 32(5):791–810, 1992.

[98] P. Milanfar. *MDSP software*, 2004.
`http://www.soe.ucsc.edu/~milanfar/SR-Software.htm, available upon request.`

[99] M. Mirmehdi, P. Clark, and J. Lam. Extracting low resolution text with an active camera for OCR. In *Proc. Spanish Symposium Pattern Recognition and Image Processing*, pages 43–48, 2001.

[100] S.K. Mitra and G.L. Sicuranza. *Nonlinear image processing*. Academic Press, 2001.

[101] M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.

[102] M. Mohri, F. Pereira, and M. Riley. Weighted automata in text and speech processing. In *Extended Finite State Models of Language: Proc. European Conf. on Artificial Intelligence*, pages 46–50, 1996.

[103] R. Mukundan, S.H. Ong, and P.A. Lee. Discrete vs. continuous orthogonal moments in image analysis. In *Proc. Int. Conf. On Imaging Systems, Science and Technology*, pages 23–29, 2001.

[104] G.K. Myers, R.C. Bolles, Q.T. Luong, and J.A. Herson. Recognition of text in 3-d scenes. In *Proc. Symposium on Document Image Understanding Technology*, pages 85–99, 2001.

[105] N. Nasios and A.G. Bors. A variational approach for color image segmentation. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 680–683, 2004.

[106] K. Negishi, M. Iwamura, S. Omachi, and H. Aso. Isolated character recognition by searching features in scene images. In *Proc. Camera-based Document Analysis and Recognition*, pages 140–148, 2005.

[107] D. Neuhoff. The Viterbi algorithm as an aid in text recognition. *IEEE Trans. Information Theory*, 21:222–226, 1975.

[108] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *Proc. Advances in Neural Information Processing Systems*, volume 14, 2001.

[109] W. Niblack. *An introduction to image processing.* Prentice-Hall, 1986.

[110] L. O'Gorman and R. Kasturi. *Document image analysis.* IEEE Computer Society Press, 1995.

[111] Y. Ojima, S. Kirigaya, and T. Wakahara. Determining optimal filters for binarization of degraded grayscale characters using genetic algorithms. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 555–559, 2005.

[112] Y. Otha. *Knowledge-based interpretation of outdoor natural scenes.* Pitman publishing, 1985.

[113] N. Otsu. A threshold selection method from gray level histograms. *IEEE Trans. System, Man and Cybernetics*, 9(1):62–66, 1979.

[114] J. Park, Y. Kwon, and J.H. Kim. An example-based prior model for text image super-resolution. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 374–378, 2005.

[115] A.J. Patti, M.I. Sezan, and A.M. Tekalp. Superresolution video reconstruction with arbitrary sampling lattices and non-zero aperture time. *IEEE Trans. Image Processing*, (8):1064–1076, 1997.

[116] T. Perroud, K. Sobottka, H. Bunke, and L. Hall. Text extraction from color documents - clustering approaches in three and four dimensions -. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 937–941, 2001.

[117] J.-P. Peters, C. Thillou, and S. Ferreira. Embedded reading device for blind people: a user-centred design. In *Proc. Emerging Technologies and Applications for Imagery Pattern Recognition*, volume 00, pages 217–222, 2004.

[118] B.T. Phong. *Illumination for computer-generated images*. PhD thesis, University of Utah, 1973.

[119] K.N. Plataniotis and A.N. Venetsanopoulos. *Color Image Processing and Applications*. Springer Verlag, 2000.

[120] S. Pollard and M. Pilu. Building cameras to capture documents. *Int. Jour. Document Analysis and Recognition*, 7(2–3), 2005.

[121] M.W. Powell and R. Murphy. Position estimation of microrovers using a spherical coordinate transform color segmenter. In *Proc. IEEE Workshop on Photometric Modeling for Computer Vision and Graphics*, pages 21–27, 1999.

[122] C. Poynton. *Color Frequently asked questions*, 1997. `http://www.inforamp.net/~poynton/notes/colour_and_gamma/ColorFAQ.html`.

[123] P. Pudil, F.J. Ferri, J. Novovicova, and J. Kittler. Floating search methods for feature selection with nonmonotonic criterion functions. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 279–283, 1994.

[124] J. Pujol, F. Martinez-Verdu, M.J. Luque, P. Capilla, and M. Vilaseca. Comparison between the number of discernible colors in a digital camera and the human eye. In *Proc. Colour, Graphics, Imaging and Vision*, pages 36–40, 2004.

[125] S.S. Raju, P.B. Pati, and A.G. Ramakrishnan. Text localization and extraction from complex color images. In *Proc. Int. Symposium Visual Computing*, pages 486–493, 2005.

[126] G. Ramponi. The rational filter for image smoothing. *IEEE Signal Processing Letters*, 3(3):63–65, 1996.

[127] D. Ruderman, T. Cronin, and C. Chiao. Statistics of cone responses to natural images: implications for visual coding. *Jour. Opt. Soc. Am. A*, 15(8):2036–2045, 1998.

[128] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. *Parallel Data Processing*, 1:318–362, 1986.

[129] B. Sankur and M. Sezgin. A survey over image thresholdýng techniques and quantitative performance evaluation. *Jour. Electronic Imaging*, 13(1):146–165, 2004.

[130] J. Sauvola and M. Pietikainen. Adaptive document image binarization. *Pattern Recognition*, 33:225–236, 2000.

[131] M. Seeger and C. Dance. Binarising camera images for OCR. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 54–58, 2001.

[132] S.A. Shafer. Using color to separate reflection components. *Color Research and Applications*, 10(4):210–218, 1985.

[133] G. Sharma. *Digital color imaging handbook*. CRC Press LLC, 2003.

[134] M. Shimizu, T. Yano, and M. Okutomi. Super-resolution under image deformation. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 586–589, 2004.

[135] M. Shin, K. Chang, and L. Tsap. Does color space transformation make any difference on skin detection ? In *Proc. IEEE Workshop on Applications of Computer Vision*, pages 275–279, 2002.

[136] W. Skarbek and A. Koschan. *Color Image Segmentation - a Survey* -, 1994. Techn. report 94-32.

[137] K. Sobottka, H. Bunke, and H. Kronenberg. Identification of text on colored book and journal covers. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 57–62, 1999.

[138] H. Stark and P. Oskoui. High resolution image recovery from image-plane arrays, using convex projections. *Jour. Optical Society of America*, 6(11):1715–1726, 1989.

[139] G.W. Stewart. On the early history of the singular value decomposition. *SIAM (Society for Industrial and Applied Mathematics) Review*, 35(4):551–566, 1993.

[140] A. Strehl. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. PhD thesis, University of Texas, 2002.

[141] J. Sun, Y. Hotta, and Y. Katsuyama. Low resolution character recognition by dual eigenspace and synthetic degraded patterns. In *Proc. ACM Hardcopy Document Processing Workshop*, pages 15–22, 2004.

[142] K. Taghva and E. Stofsky. OCRSpell: an interactive spelling correction system for OCR errors in text. *Int. Jour. Document Analysis and Recognition*, 3(3):125–137, 2001.

[143] C. Thillou, S. Ferreira, and B. Gosselin. An embedded application for degraded text recognition. *Eurasip Jour. on Applied Signal Processing, Special Issue on Advances in Intelligent Vision Systems: methods and applications*, 13:2127–2135, 2005.

[144] C. Thillou and B. Gosselin. Color binarization for complex camera-based images. In *Proc. Electronic Imaging Conf. of the Int. Society for Optical Imaging*, volume 5667, pages 301–308, 2004.

[145] C. Thillou and B. Gosselin. Combination of binarization and character segmentation using color information. In *Proc. IEEE Symposium on Signal Processing and Information Technology*, 2004.

[146] C. Thillou and B. Gosselin. Robust thresholding based on wavelets and thinning algorithms for degraded camera images. In *Proc. IEEE Advanced Concepts for Intelligent Vision Systems*, 2004.

[147] C. Thillou and B. Gosselin. Segmentation-based binarization for color degraded images. In *Proc. Int. Conf. on Computer Vision and Graphics*, 2004.

[148] R.Y. Tsai and T.S. Huang. *Multiple frame image restoration and registration. Advances in Computer Vision and Image Processing.* JAI Press Inc., 1984.

[149] Z. Tu, X. Chen, A.L. Yuille, and S.-C. Zhu. Image parsing: unifying segmentation, detection, and recognition. In *Proc. IEEE Int. Conf. Computer Vision*, pages 18–25, 2003.

[150] S. Uchida, H. Miyasaki, and H. Sakoe. Mosaicing-by-recognition for recognizing texts captured in multiple video frames. In *Proc. Camera-based Document Analysis and Recognition*, pages 3–9, 2005.

[151] J.D. van Ouwerkerk. Image super-resolution survey. *Image and Vision Computing*, 24:1039–1052, 2006.

[152] C.J. van Rijsbergen. *Information Retrieval.* 2nd edition edn. Butterworth, 1979.

[153] N. Vandenbroucke. *Segmentation d'images couleur par classification de pixels dans des espaces d'attributs colorimétriques adaptés. Application à l'analyse d'images de football.* PhD thesis, University of Lille, 2000.

[154] T.P. Vogl, J.K. Mangis, A.K. Rigler, W.T. Zink, and D.L. Alkon. Accelerating the convergence of the backpropagation method. *Biological Cybernetics*, 59:257–263, 1988.

[155] B. Wang. Minimum entropy approach to word segmentation problems. *Physica A*, 293(3):583–591, 2001.

[156] B. Wang, X.-F. Li, F. Liu, and F.-Q. Hu. Color text image binarization based on binary texture analysis. In *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pages 585–588, 2004.

[157] K. Wang and J.A. Kangas. Character location in scene images from digital camera. *Pattern Recognition*, 36:2287–2299, 2003.

[158] S. Wesolkowski. Color image edge detection and segmentation: a comparison of the vector angle and the euclidean distance color similarity measures. Master's thesis, University of Waterloo, 1999.

[159] C. Wolf, J. Jolion, and F. Chassaing. Text localization, enhancement and binarization in multimedia documents. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 1040–1057, 2002.

[160] E.K. Wong and M. Chen. A new robust algorithm for video text extraction. *Pattern Recognition*, 36(6):1397–1406, 2003.

[161] W. Wu, X. Chen, and J. Yang. Incremental detection of text on road signs from video with application to a driving assistant system. In *Proc. ACM Multimedia*, pages 10–16, 2004.

[162] G. Wyszecki and W.S. Stiles. *Color science: concepts and methods, quantitative data and formulae.* John Wiley & Sons, 1982.

[163] M. Yokobayashi and T.Wakahara. Segmentation and recognition of characters in scene images using selective binarization in color space and GAT correlation. In *Proc. Int. Conf. Document Analysis and Recognition*, pages 167–171, 2005.

[164] A. Zandifar, R. Duraiswami, and L.S. Davis. A video-based framework for the analysis of presentations/posters. *Int. Jour. Document Analysis and Recognition*, 7(2–3):178–187, 2005.

[165] J. Zhang, X. Chen, A. Hanneman, J. Yang, and A. Waibel. A robust approach for recognition of text embedded in natural scenes. In *Proc. IEEE Int. Conf. Pattern Recognition*, 2002.

[166] W. Zhao, H. Sawhney, M. Hansen, and S. Samarasekera. Super-fusion: a super-resolution method based on fusion. In *Proc. IEEE Int. Conf. Pattern Recognition*, pages 269–272, 2002.

[167] J. Zhou, D. Lopresti, and Z. Lei. OCR for world wide web images. In *Proc. SPIE on Document Recognition V*, pages 58–66, 1997.

— APPENDIX A —

Color Spaces Conversion

This appendix details conversions and visualisation[1] of color spaces described in Chapter 2 for the D65 white point[2], the 2° observer and the sRGB working space for RGB color space[3]:

- RGB

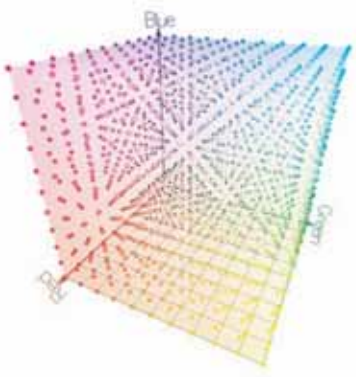


Figure A.1: RGB cube in the RGB color space.

[1] Visualization has been done with the ColorSpace software available at http://www.couleur.org/.

[2] $X_0 = 0.9504$, $Y_0 = 1.0$, $Z_0 = 1.0889$.

[3] For more details about device-dependent color spaces, the reader may refer to [78].

- RGB ⇛ CIE XYZ

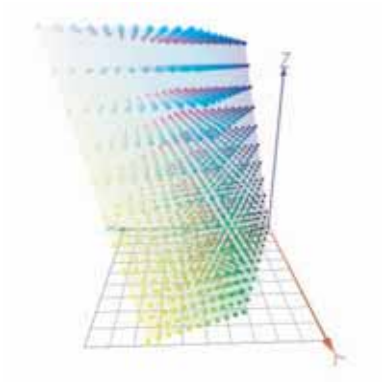


Figure A.2: RGB cube in the CIE XYZ color space.

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.412 & 0.358 & 0.180 \\ 0.213 & 0.715 & 0.072 \\ 0.019 & 0.119 & 0.0950 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

- CIE XYZ ⇛ CIE $L^*a^*b^*$

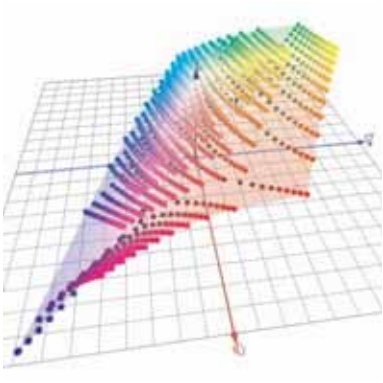


Figure A.3: RGB cube in the CIE $L^*a^*b^*$ color space.

$$\begin{cases} L^* & = & 116(Y/Y_0)^{1/3} - 16 \\ & & \text{if } Y/Y_0 > 0.008856 \\ L^* & = & 903.3(Y/Y_0) \\ & & \text{if } Y/Y_0 \leq 0.008856 \\ a^* & = & 500[f(X/X_0) - f(Y/Y_0)] \\ b^* & = & 200[f(X/X_0) - f(Y/Y_0)] \end{cases}$$

with

$$\begin{cases} f(U) & = & U^{1/3} & \text{if } U > 0.008856 \\ f(U) & = & 7.787U + 16/116 & \text{if } U \leq 0.008856 \end{cases}$$

- CIE XYZ $\Rrightarrow$ CIE $L^*u^*v^*$

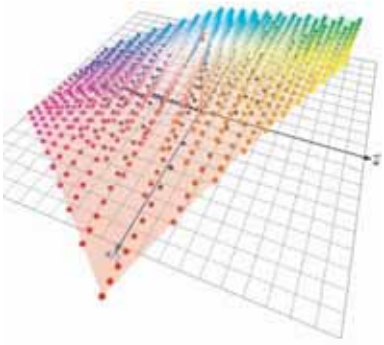


Figure A.4: RGB cube in the CIE $L^*u^*v^*$ color space.

$$\begin{cases} L^* & = & 116(Y/Y_0)^{1/3} - 16 \\ & & \text{if } Y/Y_0 > 0.008856 \\ L^* & = & 903.3(Y/Y_0) \\ & & \text{if } Y/Y_0 \leq 0.008856 \\ u^* & = & 13L^*[U(X,Y,Z) - U(X_0,Y_0,Z_0)] \\ v^* & = & 13L^*[V(X,Y,Z) - V(X_0,Y_0,Z_0)] \end{cases}$$

with

$$\begin{cases} U(X,Y,Z) & = & \frac{4X}{X+15Y+3Z} \\ V(X,Y,Z) & = & \frac{9Y}{X+15Y+3Z} \end{cases}$$

- CIE $L^*a^*b^* \Rrightarrow$ CIE L^*CH°

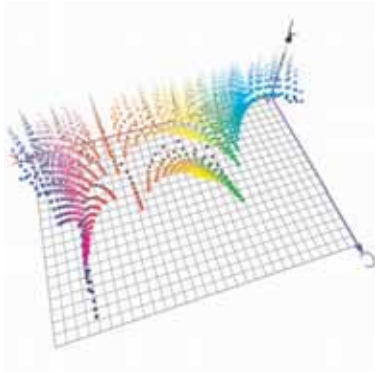


Figure A.5: RGB cube in the CIE L^*CH° color space.

$$\begin{cases} L^* & = & L^* \\ C & = & \sqrt{a^{*^2} + b^{*^2}} \\ H^\circ & = & 0 \qquad\qquad \text{whether } a^* = 0 \\ H^\circ & = & \frac{180}{\Pi}(\Pi + \arctan(\frac{b^*}{a^*})) \end{cases}$$

- RGB $\Rrightarrow$ HSI

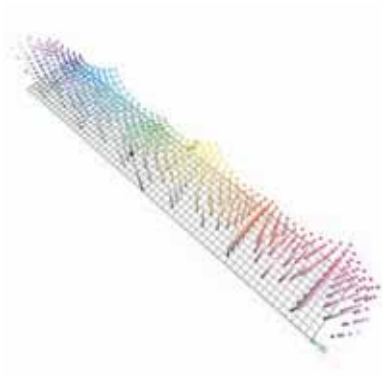


Figure A.6: RGB cube in the HSI color space.

$$\begin{cases} H & = & \arctan(\beta/\alpha) \\ S & = & \sqrt{\alpha^2+\beta^2} \\ I & = & (R+G+B)/3 \end{cases}$$

with

$$\begin{cases} \alpha & = & R-\frac{1}{2}(G+B) \\ \beta & = & \frac{\sqrt{3}}{2}(G-B) \end{cases}$$

- RGB $\Rrightarrow$ HSV

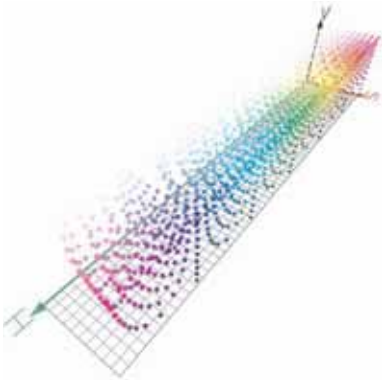


Figure A.7: RGB cube in the HSV color space.

$$\begin{cases} H & = & 0.235(G-B)/(R-min(R,G,B)) \\ & & \text{if } max(R,G,B)=R \\ H & = & 0.235(B-R)/(G-min(R,G,B)) \\ & & \text{if } max(R,G,B)=G \\ H & = & 0.235(R-G)/(B-min(R,G,B)) \\ & & \text{if } max(R,G,B)=B \\ S & = & \frac{max(R,G,B)-min(R,G,B)}{max(R,G,B)} \\ V & = & max(R,G,B) \end{cases}$$

- RGB $\Rrightarrow I_1I_2I_3$

$$\begin{cases} I_1 & = & \frac{1}{3}(R+G+B) \\ I_2 & = & \frac{1}{2}(R-B) \\ I_3 & = & \frac{1}{4}(2G-R-B) \end{cases}$$

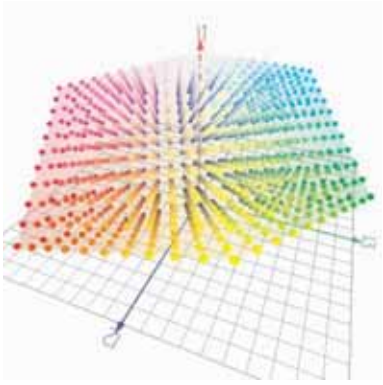


Figure A.8: RGB cube in the $I_1I_2I_3$ color space.

- RGB $\Rrightarrow$ CMYK

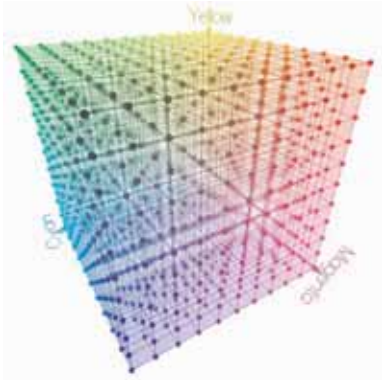


Figure A.9: RGB cube in the CMY color space. K stands for Key and corresponds to the additional black ink added for printing.

$$\begin{cases} C & = & 1-R \\ M & = & 1-G \\ Y & = & 1-B \\ K & = & min(C,M,Y) \end{cases}$$

- RGB ⇛ YIQ

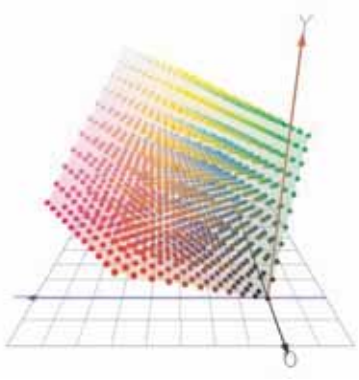


Figure A.10: RGB cube in the YIQ color space.

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.115 \\ 0.596 & -0.274 & -0.322 \\ 0.212 & -0.523 & 0.311 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

- RGB ⇛ YUV

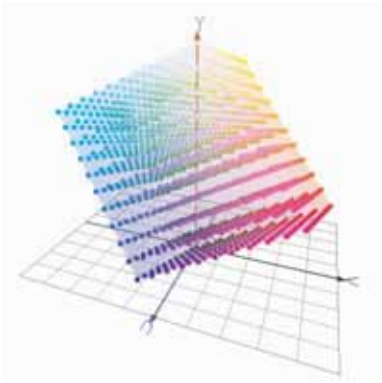


Figure A.11: RGB cube in the YUV color space.

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.115 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

- RGB $\Rrightarrow YCbCr$

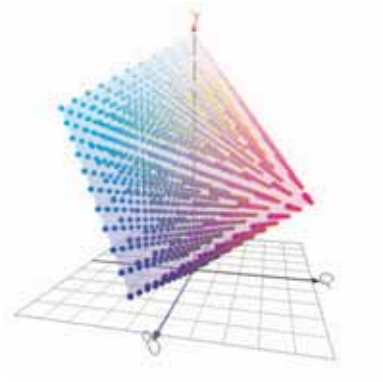


Figure A.12: RGB cube in the $YCbCr$ color space.

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.115 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.418 & -0.082 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

— APPENDIX B —

Expectation-Maximization

This appendix describes the Expectation-Maximization (EM) algorithm in the framework of Gaussian mixture modelling and data clustering:

- Consider N points $\mathcal{X} = (X_1 \dots X_N)$ and each $X_j = (X_{j,1} \dots X_{j,d})$ from a d-dimensional vector space. Here, $d = 3$ for RGB color space.
- Clusters are represented by mixture of Gaussian distributions.
- Representation of a cluster i (with one Gaussian representing one cluster): center points μ_i of all points in the cluster and $d \times d$ covariance matrix Σ_i for the points in the cluster i.
- Density function g_i for cluster i:

$$g_i(\mathcal{X}|\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\Pi)^d \det(\Sigma_i)}} \exp^{\left(-\frac{1}{2}(\mathcal{X}-\mu_i)^T \Sigma_i^{-1}(\mathcal{X}-\mu_i)\right)} \tag{B.1}$$

  where det is the determinant of the covariance matrix Σ_i.

- Let p_i denote the fraction of clusters i in the entire data set D for the probabilistic model $P(\mathcal{X})$:

$$P(\mathcal{X}) = \sum_{i=1}^{M} p_i g_i(\mathcal{X}|\mu_i, \Sigma_i) \text{ with } \sum_{i=1}^{M} p_i = 1 \tag{B.2}$$

  where M is the number of clusters.

For univariate case (one Gaussian only), it is possible to solve parameters of the Gaussian with the Maximum Likelihood Estimation method. For multivariate case, the EM algorithm is traditionally used due to the presence of hidden variables. For each observable point X, the hidden variable indicates to which cluster it belongs to. EM is the solution to the chicken-and-egg problem and is mainly used in the case of HMM or GMM, which is our concern in this appendix. It iterates two steps being E and M, standing for Expectation and Maximization respectively. The main goal is to maximize the likelihood function, which represents the probability of data given the model, such as described in the following equation:

$$P(data|model) = P(\mathcal{X}|\mu, \Sigma) = \prod_{j=1}^{N} \sum_{i=1}^{M} p_i P(X_j|\mu_i, \Sigma_i) \quad \text{(B.3)}$$

Maximization of the likelihood is a measure for the quality of the clustering, meaning how well data fit the model. To assign points to clusters, the Bayes rule may be considered to compute probabilities. A point X may belong to several clusters with different probabilities and the posterior probability is given by:

$$P(\mu_i, \Sigma_i|X) = p_i \frac{g_i(X|\mu_i, \Sigma_i)}{\sum_{i=1}^{M} p_i g_i(X|\mu_i, \Sigma_i)} \quad \text{(B.4)}$$

The EM procedure is as following:

1. Initialize parameters $\mu_{i,n}$, $\Sigma_{i,n}$, $p_{i,n}$ where n is the number of iterations. At the first iteration, $n = 1$.

2. E step: Compute $P(X)|\mu_{i,n}, \Sigma_{i,n})$, $P(X)$ and hence $P(\mu_{i,n}, \Sigma_{i,n}|X)$ for each object X from the data set D and each cluster i

3. M step: Recompute the model by calculating $p_{i,n+1}$, $\mu_{i,n+1}$ and $\Sigma_{i,n+1}$ with

$$p_{i,n+1} = \frac{1}{N} \sum_{X \in D} P(\mu_{i,n}, \Sigma_{i,n}|X) \quad \text{(B.5)}$$

$$\mu_{i,n+1} = \frac{\sum_{X \in D} XP(\mu_{i,n}, \Sigma_{i,n}|X)}{\sum_{X \in D} P(\mu_{i,n}, \Sigma_{i,n}|X)} \quad \text{(B.6)}$$

$$\Sigma_{i,n+1} = \frac{\sum_{X \in D} P(\mu_{i,n}, \Sigma_{i,n}|X)(X - \mu_{i,n})^2}{\sum_{X \in D} P(\mu_{i,n}, \Sigma_{i,n}|X)} \tag{B.7}$$

Steps 2 and 3 are iterated until:

$$|P(data|model_n) - P(data|model_{n+1})| < \varepsilon \tag{B.8}$$

The convergence is ensured but a possible local minimum is not discarded. The first step is responsible of the initialization sensitivity of the EM algorithm, leading to different results. Several papers propose to circumvent this issue, such as Figueiredo and Jain [36]. More details on proofs of the EM algorithm may be found in the explanation of Dempster et al. [21], which popularized the algorithm in 1977.