TEXT EXTRACTION FROM NATURAL SCENE: METHODOLOGY AND APPLICATION

by

CHUCAI YI

Submitted to the Department of Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Science
at
THE GRADUATE CENTER, CITY UNIVERSITY OF NEW YORK
2014

Doctoral Committee:

Professor	Ying-Li Tian, Chair
Professor	Zhigang Zhu
Professor	Andrew Rosenberg
Professor	Fred Moshary
Dr.	Liangliang Cao

ABSTRACT

With the popularity of the Internet and the smart mobile device, there is an increasing demand for the techniques and applications of image/video-based analytics and information retrieval. Most of these applications can benefit from text information extraction in natural scene. However, scene text extraction is a challenging problem to be solved, due to cluttered background of natural scene and multiple patterns of scene text itself. To solve these problems, this dissertation proposes a framework of scene text extraction.

Scene text extraction in our framework is divided into two components, detection and recognition. Scene text detection is to find out the regions containing text from camera captured images/videos. Text layout analysis based on gradient and color analysis is performed to extract candidates of text strings from cluttered background in natural scene. Then text structural analysis is performed to design effective text structural features for distinguishing text from non-text outliers among the candidates of text strings. Scene text recognition is to transform image-based text in detected regions into readable text codes. The most basic and significant step in text recognition is scene text character (STC) prediction, which is multi-class classification among a set of text character categories. We design robust and discriminative feature representations for STC structure, by integrating multiple feature descriptors, coding/pooling schemes, and learning models. Experimental results in benchmark datasets demonstrate the effectiveness and robustness of our proposed framework, which obtains better performance than previously published methods.

Our proposed scene text extraction framework is applied to 4 scenarios, 1) reading print labels in grocery package for hand-held object recognition; 2) combining with car detection to localize license plate in camera captured natural scene image; 3) reading indicative signage for assistant navigation in indoor environments; and 4) combining with object tracking to perform scene text extraction in video-based natural scene. The proposed prototype systems and associated evaluation results show that our framework is able to solve the challenges in real applications.

ACKNOWLEDGEMENTS

First I would like to thank my advisor, Prof. Yingli Tian, who offered me a valuable opportunity of PhD study in the field of computer vision. It is Prof. Tian's advice, support and patience that help me grow up into a researcher. I would also like to thank all my committee members. I learned a lot through the collaborations on research projects or the discussions in their fantastic lectures. They provided many constructive comments and insightful suggestions for my dissertation.

I am also very thankful to my collaborators during my PhD study. I would like to thank Prof. Zhigang Zhu for the collaboration on the research project of Multimodal and Alternative Perception for Visually Impaired People. His suggestion was always constructive and significant to my work. I would like to thank Prof. Jizhong Xiao and Dr. Aries Arditi for helpful discussions and suggestions on the design of Intelligent Situation Awareness and Navigation Aid for Visually Impaired Persons, as well as the collaboration on some papers. I would like to thank Xiaodong Yang, Dr. Liangliang Cao from IBM Research Watson and Chenyang Zhang for the collaboration on Surveillance Event Detection in TRECVID 2012. I would like to thank Dr. Lin Luo and Qinying Liao from IBM Research China for the opportunity of summer intern in IBM Research China in 2013. We worked together on Social Media Data Analytics. I would like to thank Dr. Shizhi Chen, Long Tian, Ze Ye, Hangrong Pan, Shuihua Wang, Robert Flores, and Samleo Joseph for the collaborations on research work and paper writing. All of them are very good teammates. Besides, I owe sincere gratitude to my collaborators (in alphabetical order) Dr. Cheng Chen, Carol Mazuera, Edgardo Molina, Michael Quintian, Xuejian Rong, Yang Xian, Shuai Yuan, Xiaochen Zhang.

In the past 5 years, I made a lot of friends and I enjoyed the life in the Great New York City. My PhD study is a very exciting and fruitful journey. At last, I would like to thank my parents for their support. I would never acquire these achievements without their love and encouragement.

Contents

Abstract....................I

Acknowledgements....................III

Contents....................V

List of Figures....................XI

List of Tables....................XVII

Chapter 1 Text Information in Natural Scene....................1

1.1. Motivations and Challenges....................1

1.2. Baseline Solutions....................3

Chapter 2 Scene Text Character Components....................5

2.1. Problem Formulation....................5

2.2. Previous Work....................5

2.3. Color-Based Partition....................8

2.3.1. Absolute Color Reduction....................8

2.3.2. Bigram Color Reduction....................11

2.4. Gradient-Based Partition....................17

2.4.1. Contour Search....................17

2.4.2. Connecting Path....................18

2.5. Two Popular Methods of Extracting Text Character Components 22

2.5.1. Maximal Stable Extremal Region 23

2.5.2. Stroke Width Transform 24

2.5.3. Comparisons with Gradient-based and Color-based Methods26

2.6. Geometrical Constraints of Candidate Character Components 26

2.7. Summary of Contributions 28

Chapter 3 Scene Text String Fragments 29

3.1. Problem Formulation 29

3.2. Previous Work 30

3.3. Adjacent Character Grouping 31

3.4. Text Line Fitting 36

3.5. Geometrical Constraints of Candidate String Fragments 40

3.6. Summary of Contributions 41

Chapter 4 Structure Modeling for Predicting True Positive Text String Fragments 43

4.1. Problem Formulation 43

4.2. Previous Work 44

4.3. Feature Representations for Text String based on Feature Maps. 45

4.3.1. Haar-like Block Patterns 45

4.3.2. Feature Maps 46

A Gradient 47

B Text Stroke Width 47

C Text Stroke Orientation 48

D Distribution of Edge Pixels 50

4.3.3. Feature Value Generation 51

4.4. Cascaded Adaboost Learning 52

4.4.1. Weak Classifiers 53

4.4.2. Inference 54

4.5. Summary of Contributions 55

Chapter 5 Structure Modeling for Scene Text Character Prediction 57

5.1. Problem Formulation 57

5.2. Previous Work 59

5.3. Feature Representations for Predicting STC Categories 60

5.3.1. Low-level Feature Descriptor 60

5.3.2. Dense Sampling and Bag-of-words Modeling 61

5.3.3. Dense Sampling and Fisher Vector 63

5.3.4. Global Sampling 65

5.4. Feature Representations for Predicting True Positive STCs from Background Outliers 66

5.5. Learning Process 69

5.6. Summary of Contributions 70

Chapter 6 Word-level Configuration of Scene Text by Conditional Random Field 71

6.1. Problem Formulation 71

6.2. Conditional Random Field 72

6.3. Summary of Contributions 74

Chapter 7 Scene Text Detection 75

7.1. Datasets 75

7.2. Evaluation Measures 76

7.3. Evaluating Scene Text Detection by Text Layout Analysis on ICDAR Datasets 77

7.4. Evaluating Scene Text Detection by Structural Feature Maps on Image Patches from ICDAR Datasets 78

7.5. Evaluating Scene Text Detection by Combing Layout Analysis and Structural Analysis 80

7.6. Evaluating Scene Text Extraction on Born-Digital Images and Video Images 83

7.7. Some Challenging situations to be addressed 84

7.8. Summary of Contributions 86

Chapter 8 Scene Text Recognition 87

8.1. Datasets and Evaluation Measures of Scene Text Character Prediction 87

8.2. Evaluating Feature Representations of Densely Local Sampling and BOW models 88

8.2.1. Evaluating Low-level Descriptors 88

8.2.2. Evaluating Dictionary Size in BOW Model 91

8.2.3. Evaluating Coding Pooling Scheme 92

8.2.4. Evaluating SVM Kernel 94

8.3. Evaluating Feature Representations of Global Sampling 96

8.3.1. Complete STC in Global Sampling 96

8.3.2. Incomplete STC in Global Sampling 96

8.4. Datasets and Evaluation Measures of Word Recognition 98

8.5. Evaluating Word Recognition 99

8.6. Summary of Contributions 100

Chapter 9 Blind-assistant Hand-Held Object Recognition 101

9.1. Blind-Assistant Prototype System 101

9.2. Implementing Scene Text Extraction in PC Platform 103

9.3. Evaluation Results and Discussions 105

9.4. Summary 108

Chapter 10 Tracking-based Scene Text Recognition 109

10.1. Scene Text Tracking 109

10.2. STC Prediction and Word-Level Configuration 111

10.3. Dataset and Evaluation Measures 111

10.4. Experimental results and discussions 112

10.5. Summary 114

Chapter 11 License Plate Detection 115

11.1. Car Detection 115

11.2. License Plate Localization 116

11.3. Post Processing 116

11.4. Evaluation Results and Discussions 118

11.5. Summary 119

Chapter 12 Indoor Navigation ... 121

12.1. Door Detection and Room Number Extraction ... 121

12.2. Floor Plan Parse for Way-finding ... 123

12.3. Implementing Scene Text Extraction in Mobile Platform ... 124

12.4. Summary ... 126

Chapter 13 Conclusions and Discussions ... 127

Appendix A. The Involved Learning Models in Our Framework ... 131

A.1 Adaptive Boost ... 131

A.2 Support Vector Machine ... 131

Appendix B. Publications During PhD Study ... 135

Reference ... 137

LIST OF FIGURES

FIG. 1–1 SOME EXAMPLES OF (A) SCAN DOCUMENT TEXT; (B) HANDWRITTEN TEXT; (C) SCENE TEXT. 2

FIG. 1–2. THE FLOWCHART OF OUR PROPOSED SCENE TEXT EXTRACTION FRAMEWORK.. 3

FIG. 2–1 (A) A SCENE IMAGE WITH MULTIPLE COLORS; (B) COLOR DISTRIBUTION IN RGB SPACE; (C) FOUR INITIAL CUBE COLOR CLUSTERS WITH RADIUS h. 10

FIG. 2–2 SOME EXAMPLES OF COLOR-BASED PARTITION, WHERE THE LEFT COLUMN CONTAINS ORIGINAL IMAGES AND OTHER COLUMNS CONTAIN THE CORRESPONDING DOMINANT COLOR LAYERS. 11

FIG. 2–3 (A) A SCENE IMAGE. (B) ATTACHMENT SURFACE OF TEXT. (C) TWO EXAMPLES OF COLOR-PAIRS CORRESPONDING TO THE BOUNDARIES AT THE SIGNAGE BOARD AND WINDOW GRID RESPECTIVELY. 12

FIG. 2–4. (A) EXAMPLES OF BOUNDARY LAYERS FROM SCENE IMAGES; EDGE PIXELS WITH SIMILAR COLOR-PAIRS AND SPATIAL POSITIONS ARE GROUPED INTO THE SAME LAYER. BOUNDARIES AT DIFFERENT VERIDICAL POSITIONS ARE ASSIGNED INTO DIFFERENT BOUNDARY LAYERS BECAUSE OF Y-COORDINATE SPATIAL INFORMATION IN CLUSTERING PROCESS. (B) RESULTS OF COLOR REDUCTION BASED ON CLUSTERING OF ABSOLUTE COLOR VALUES, WHERE WHITE REGION REPRESENTS BACKGROUND, AND COLOR REGION CONSISTS OF THE PIXELS THAT ARE GROUPED TO THE LAYER. 16

FIG. 2–5. (A) CANNY EDGE MAP OF A SCENE IMAGES. (B) BOUNDING BOXES OF ALL AVAILABLE EDGE BOUNDARIES IN THE FORM OF CONNECTED EDGE PIXELS, OBTAINED FROM EDGE MAP. 18

FIG. 2–6 (A) EXAMPLES OF PIXEL COUPLES; (B) CONNECTING PATHS OF ALL PIXEL COUPLES ARE MARKED AS WHITE FOREGROUND WHILE OTHER PIXELS ARE MARKED AS BLACK BACKGROUND. 19

FIG. 2–7. (A) TWO CONNECTING PATHS OF PIXEL COUPLES MARKED BY BLUE AND RED CIRCLES RESPECTIVELY. (B) THE CORRESPONDING EXPONENTIAL DISTRIBUTION OF GRADIENT MAGNITUDES ON THE CONNECTING PATHS. (C) PARTITIONED COMPONENTS OBTAINED FROM THE FIRST ROUND. 21

FIG. 2–8. CONNECTING PATH OF PIXEL COUPLE ALONG THE LARGER SIDE OF RECTANGLE STROKE IS ANALYZED IN THE SECOND ROUND PARTITION. TOP ROW SHOWS PIXEL COUPLES IN PURPLE ACROSS THE LARGER SIDE OF RECTANGLE STROKES. BOTTOM ROW PRESENTS THE PARTITIONED COMPONENTS OBTAINED FROM THE FIRST ROUND AND THE SECOND ROUND RESPECTIVELY. 21

FIG. 2–9. CONNECTED COMPONENTS OBTAINED FROM DIRECT MORPHOLOGICAL PROCESSING ON GRAY IMAGES AND CORRESPONDING BINARY IMAGES. WE COMPARE RESULTS OF FOUR MORPHOLOGICAL OPERATORS WITH RESULT OF OUR GRADIENT-BASED PARTITION. 22

FIG. 2–10. (A) ORIGINAL NATURAL SCENE IMAGE. (B) CANNY EDGE MAP. (C) SOME MSER REGIONS AS WHITE FOREGROUND. 23

FIG. 2–11. STROKE WIDTH TRANSFORM: A CHARACTER STROKE IS REPRESENTED BY THE RED BOUNDARY. THE TWO PIXELS p AND q ARE ON THE BOUNDARY OF A CHARACTER STROKE. STROKE WIDTH IS CALCULATED AS THE DISTANCE BETWEEN two pixels which have similar gradient magnitude and opposite gradient direction. Each pixel is assigned a value of stroke width to be a stroke width map, and the pixels with similar values in this map are grouped together to compose the connected components on the right, which will be used as candidate character component in later processes. 25

Fig. 3–1. (a) Sibling group of the connected component 'r' where 'B' comes from the left sibling set and 'o' comes from the right sibling set; (b) Merge the sibling groups into an adjacent character group corresponding to the text string "Brolly?"; (c) Two detected adjacent character groups marked in red and green respectively. 32

Fig. 3–2. (a) Centroids of connected components in a color layer; (b) $D(mA, mB)$ approximately equals to $D(mB, mC)$ in text region while $D(mA, mP)$ is much larger than $D(mP, mQ)$ in background, where $D(.)$ represents Euclidean distance; (c) Three neighboring connected components in red share similar areas while those in green have very different areas; (d) Resulting fitted lines from centroids cascading. Red line corresponds to text region while cyan lines are false positives to be removed. 38

Fig. 4–1. Some examples of text fragment samples in the form of image patches. 44

Fig. 4–2. Diagram of the proposed Adaboost learning based text region localization algorithm by using stroke orientations and edge distributions. 44

Fig. 4–3. Some examples of Haar-like block patterns to extract text structural features. Features are obtained by the absolute value of mean pixel values in white regions minus those in black regions. 46

Fig. 4–4. From top to bottom, the figures represent the original text patch, the feature maps of gradient magnitude and the feature maps of gradient orientation respectively. 47

Fig. 4–5. A fragment sample of text showing relationships between stroke orientations and gradient orientations at pixels of stroke boundaries. Blue arrows denote the stroke orientations at the sections and red arrows denote the gradient orientations at pixels of stroke boundaries. 48

Fig. 4–6. (a) An example of stroke orientation label. The pixels denoted by blue points are assigned the gradient orientations (red arrows) at their nearest edge pixels, denoted by the red points. (b) A 210×54 fragment sample and its 16-bin histogram of quantized stroke orientations. 50

Fig. 4–7. The flowchart of cascaded Adaboost classifier. It consists of a sequence of stage-Adaboost classifier. Each stage-Adaboost classifier combines an optimized set of weak classifiers. The set of negative fragment samples in training process keeps updated stage-by-stage, ensuring that a negative cannot be correctly classified by all previous stages. 56

Fig. 5–1. Flowchart of STC prediction framework in performance evaluation, where the shadowed boxes denote the processing steps with multiple options and the white boxes denote the outputs of the

PROCESSING STEPS. THE BLUE ARROWS DENOTE THE PROCESSING OF LOCAL SAMPLING BASED FEATURE REPRESENTATION, AND THE RED ARROWS DENOTE THE PROCESSING OF GLOBAL SAMPLING BASED FEATURE REPRESENTATION. 58

FIG. 5–2. DETECTED INTEREST POINTS IN RED AND THEIR NEIGHBORING WINDOWS IN CYAN. THE LEFT IS OBTAINED FROM MULTI-SCALE SALIENCY DETECTOR AND THE RIGHT IS OBTAINED FROM HARRIS CORNER DETECTOR........................... 67

FIG. 5–3. TWO STRUCTURE CORRELATION EXAMPLES. THE CYAN SQUARES REPRESENT ORIGINAL DETECTED INTEREST POINT WINDOWS, AND THE RED SQUARES REPRESENT THEIR CORRESPONDING SYMMETRIC COUNTERPARTS. WE EXTRACT THE ORIGINAL WINDOWS AND ONE OF THEIR SYMMETRIC COUNTERPARTS (MARKED BY RED DASH LINE), AND CALCULATE THEIR CORRELATED WINDOWS UNDER ABSOLUTE DIFFERENCE AND RECIPROCAL OF SUM. 68

FIG. 6–1. TOP ROW DEMONSTRATES THE WORD RECOGNITION BASED ON PERFECT STC PREDICTIONS, AND BOTTOM ROW DEMONSTRATES THE ERROR OF WORD RECOGNITION BECAUSE OF SLIGHT ERROR OF STC PREDICTIONS...................... 71

FIG. 6–2. ON THE LEFT, STC IS EXTRACTED BY DETECTION AND TRACKING, AND THEN TRANSFORMED INTO A FEATURE REPRESENTATION THROUGH LOW-LEVEL FEATURE DESCRIPTOR AND CODING & POOLING PROCESSES. SVM-BASED STC PREDICTOR IS APPLIED TO OBTAIN ITS CATEGORY LABEL AND PREDICTION SCORE. ON THE RIGHT, IN A TRACKING FRAME OF SCENE TEXT, EACH STC BOUNDING BOX IS DEFINED AS A NODE IN CRF MODEL. UNARY COST OF STC PREDICTION AND PAIRWISE COST OF BIGRAM LEXICON IS DEFINED IN THIS GRAPHICAL MODEL. 73

FIG. 7–1. EVALUATION RESULTS OF THE 5 GABOR-BASED FEATURES ON ICDAR-2003 ROBUST READING DATASET OF SCENE IMAGES. HERE HIT RATE IS THE RATIO OF CORRECTLY CLASSIFIED SAMPLES IN POSITIVE SET, AND FALSE POSITIVE RATE IS THE RATIO OF INCORRECTLY CLASSIFIED SAMPLES IN NEGATIVE SET. 79

FIG. 7–2. EXAMPLE RESULTS OF TEXT LOCALIZATION IN THE ICDAR-2003 DATASET, WHERE THE TEXT REGIONS ARE MARKED BY CYAN BOXES.. 81

FIG. 7–3. SOME EXAMPLE RESULTS OF TEXT LOCALIZATION IN THE ICDAR-2011 DATASET, WHERE THE TEXT REGIONS ARE MARKED BY CYAN BOXES.. 82

FIG. 7–4. EXAMPLE RESULTS OF TEXT LOCALIZATION IN THE BORN-DIGITAL IMAGES. 84

FIG. 7–5. EXAMPLE RESULTS OF TEXT LOCALIZATION IN BROADCAST VIDEO IMAGES. 84

FIG. 7–6. EXAMPLES OF IMAGES WHERE OUR METHOD FAILS. 85

FIG. 8–1. SOME EXAMPLES OF STCS CROPPED FROM TEXT REGIONS. MOST STCS HAVE SIMILAR STRUCTURE TO ANOTHER COUNTERPARTS. 88

FIG. 8–2. PERFORMANCE EVALUATIONS OF 6 TYPES OF FEATURE DESCRIPTORS UNDER THREE CODING/POOLING SCHEMES, DICTIONARY SIZE 2000 AND CHI-SQUARE SVM KERNEL. THE VALUE -0.1 IN HORIZONTAL AXIS DENOTES β VALUE IN EQ. (5–1). THE TOP FIGURE SHOWS RESULTS FROM CHARS74K AND THE BOTTOM FIGURE SHOWS RESULTS FROM ICDAR2003CH.. 89

FIG. 8–3. EVALUATION RESULTS OF TEXT CLASSIFICATION. TEXT DESCRIPTORS DERIVED FROM STRUCTURE CORRELATION PERFORM BETTER THAN THOSE DIRECTLY DERIVED FROM INTEREST POINT WINDOWS. IN ADDITION, TEXT DESCRIPTOR

PRESENTED BY LEFT FIGURE IS CALCULATED FROM APPEARANCE FEATURES UNDER HARRIS-CORNER DETECTOR AND HOG DESCRIPTOR, AND THE RIGHT ONE IS OBTAINED FROM APPEARANCE FEATURES UNDER HARRIS-CORNER DETECTOR AND SIFT DESCRIPTOR. 90

FIG. 8–4. PERFORMANCE EVALUATIONS OF 5 DICTIONARY SIZES UNDER FOUR FEATURE DESCRIPTORS AND CHI-SQUARE SVM KERNEL. TOP FIGURE RESULTS FROM CHARS74K AND BOTTOM FIGURE RESULTS FROM ICDAR2003CH. 91

FIG. 8–5. EVALUATING 6 CODING/POOLING SCHEMES, UNDER THREE FEATURE DESCRIPTORS, DICTIONARY SIZE 1000, AND CHI-SQUARE SVM KERNEL. TOP FIGURE IS FROM CHARS74K AND BOTTOM FIGURE IS FROM ICDAR2003CH. 93

FIG. 8–6. STC PREDICTION RESULTS IN CHARS74K, INCLUDING ALL KINDS OF CODING AND POOLING SCHEMES. FISHER VECTOR OBTAINS BETTER PERFORMANCE THAN THE OTHERS. 94

FIG. 8–7. EVALUATING LINEAR KERNEL AND $\chi 2$ KERNEL IN SVM. TOP FIGURE IS FROM CHARS74K AND BOTTOM FIGURE IS FROM ICDAR2003CH. 95

FIG. 8–8. PERFORMANCE EVALUATIONS BASED ON GHOG GLOBAL SAMPLING. THE LEFT FIRST BAR DENOTES THE ACCURATE RATE OF GLOBAL HOG IN COMPLETE AND ORIGINAL PATCH. BLUE BARS AND YELLOW BARS DENOTE ACCURACY RATE OF INCOMPLETE PATCHES AS THEIR TOP EXAMPLES. THE FIRST RED BAR DENOTES ACCURACY RATE (0.63) OF PREPROCESSED PATCHES BY GAUSSIAN AND THE SECOND RED BAR DENOTES THAT OF LAPLACIAN FILTERS (0.46). 97

FIG. 9–1. TWO EXAMPLES OF EXTRACTING TEXT BY THE PROTOTYPE SYSTEM FROM CAMERA-CAPTURED IMAGES. TOP: A MILK BOX; BOTTOM: A MEN BATHROOM SIGNAGE. (A) CAMERA-CAPTURED IMAGES; (B) LOCALIZED TEXT REGIONS (MARKED IN BLUE); (C) TEXT REGIONS CROPPED FROM IMAGE; (D) TEXT CODES RECOGNIZED BY OCR. THE TOP-RIGHT PORTION OF THE BOTTOM IMAGE THAT CONTAINS TEXT IS ALSO SHOWN IN A MAGNIFIED CALLOUT, FOR CLARITY. 102

FIG. 9–2. FLOWCHART OF OUR PROTOTYPE SYSTEM TO READ TEXT FROM HAND-HELD OBJECTS FOR BLIND USERS. 103

FIG. 9–3. A SNAPSHOT OF OUR PROTOTYPE SYSTEM, INCLUDING THREE FUNCTIONAL COMPONENTS FOR SCENE CAPTURE, DATA PROCESSING AND AUDIO OUTPUT. 104

FIG. 9–4. PROTOTYPE SYSTEM ASSISTS BLIND USER READ TEXT INFORMATION FROM HAND-HELD OBJECTS, INCLUDING THE DETECTED TEXT REGIONS IN CYAN AND THE RECOGNIZED TEXT CODES. 105

FIG. 9–5. EXAMPLES OF BLIND PERSONS ARE CAPTURING IMAGES OF THE OBJECT IN THEIR HANDS. 106

FIG. 9–6. (A) SOME RESULTS OF TEXT DETECTION ON THE USER-CAPTURED DATASET, WHERE LOCALIZED TEXT REGIONS ARE MARKED IN BLUE. (B) TWO GROUPS OF ENLARGED TEXT REGIONS, BINARIZED TEXT REGIONS, AND WORD RECOGNITION RESULTS FROM TOP TO DOWN. 107

FIG. 10–1. INITIAL TEXT REGIONS ARE EXTRACTED BY GROUPING ADJACENT CONNECTED COMPONENTS IN SIMILAR SIZE AND HORIZONTAL ALIGNMENT IN THE MSER MAP. EACH SCENE TEXT CHARACTER WILL BE INDEPENDENTLY TRACKED WITH THE MULTI-OBJECT TRACKING METHOD. A TRAJECTORY IS THEN ESTIMATED BY MERGING THE TRACKED STC BOUNDING BOX IN EACH FRAME. THE STC PREDICTION SCORES ALONG THIS STC TRAJECTORY WILL BE USED TO IMPROVE SCENE TEXT RECOGNITION. 110

Fig. 10–2. Some example results of word recognition from multiple frames of text tracking. Right column shows recognized words. 114

Fig. 11–1. Some examples of car-rear training samples. Each car is located at complex background. The LPs are mostly located in the central parts of car rear. 115

Fig. 11–2. (a) Original image of license plate. (b) Edge map from canny edge detection. (c) Detected car region in red box and detected text regions in cyan rectangle boxes. (d) Post-processing is applied to filter out background outliers and preserve the region with the highest edge density as ultimate LP region. 117

Fig. 11–3. Some post-processing examples in LP detection. Cyan regions inside red car boxes represent the true positive LP, and red regions inside red car boxes represent false positive LP detections that are filtered out by post processing. 118

Fig. 11–4. (a) LPs are partially detected. (b) Some LPs are missing because of severe illuminations or low resolutions. 118

Fig. 12–1. (a) Edges and corners are used for door detection. (b) Door detection under cluttered background. 122

Fig. 12–2. Door detection and further scene text extraction from doors. This is a combination of door detection and scene text extraction techniques for generating current locations. 122

Fig. 12–3. (a) An example of floor plan, where the blue shaded region will be analyzed. (b) Each room number is regarded as a node, and a way from room "632" to room "623" is marked in yellow. (c) The abstract graph of the floor plan map, where the yellow edge indicates an edge from node "632" to node "623", corresponding the yellow way in (b). 124

Fig. 12–4. Our prototype system of scene text extraction in Android platform. The text strings "GARMIN", "FIRE", "EXIT", and "Cheerios" are extracted from complex background. Although some characters are incorrectly recognized in current prototype, we will introduce lexicon analysis to improve the recognition performance. 125

LIST OF TABLES

TABLE 3–1. THE PROCEDURE OF ADJACENT CHARACTER GROUPING 35

TABLE 3–2. THE PROCEDURE OF TEXT LINE FITTING 39

TABLE 7–1. THE COMPARISON BETWEEN OUR ALGORITHM AND THE TEXT DETECTION ALGORITHMS PRESENTED IN [**51**] [**50**] ON THE ROBUST READING DATASET. 78

TABLE 7–2. THE COMPARISON BETWEEN OUR FRAMEWORK AND THE TEXT LOCALIZATION ALGORITHMS PRESENTED IN [**22**] [**51**] [**50**] ON THE ROBUST READING DATASET. 80

TABLE 7–3. THE RESULTS OF ICDAR-2011 ROBUST READING COMPETITION ON SCENE TEXT LOCALIZATION (%) [**72**]. OUR PROPOSED FRAMEWORK WON 2ND PLACE. 82

TABLE 8–1. COMPARATIONS BETWEEN OUR BEST RESULTS AND EXISTING METHODS OF STC PREDICTION OVER THE BENCHMARK DATASETS. 98

TABLE 8–2. THE EXPERIMENTAL RESULTS OF WORD RECOGNITION IN ICDAR 2011, WHICH ARE EVALUATED BY ED0 AND ED1. 99

TABLE 10–1. THE RESULTS OF WORD RECOGNITION IN A SINGLE FRAME AND MULTIPLE FRAMES OF TEXT RACKING. 113

Chapter 1 TEXT INFORMATION IN NATURAL SCENE

1.1. MOTIVATIONS AND CHALLENGES

Text has been an effective tool for broadcasting information and exchanging ideas for thousands of years. When it comes to text in image, we may first think of scan document or camera-based hand-written paper (see Fig. 1–1 (a-b)). However, text widely exists in natural scene, mostly in the form of informative signage like shop sign and traffic sign. Text information plays an important role in our everyday life, because it provides the most straightforward and unambiguous explanations of surrounding environments. Nowadays with the popularity of smart mobile device, there is an increasing demand for the image/video-based applications such as reading aid, assistive navigation, content-based image retrieval, and geocoding. Almost all these applications can benefit from automatic text information extraction from natural scene. Since the promising prospect in real applications and the challenging problems to be addressed, scene text extraction is chosen to be my dissertation topic. I will present a complete framework of text information extraction from camera-based natural scene. This framework involves many state-of-the-art techniques and models in the fields of image processing, computer vision, and machine learning. Also we design novel algorithms to solve the common challenging problems in scene text extraction.

Many research work of scene text extraction has been published, but it is still an opening research topic. It is a challenging task to extract text information from natural scene images for four main reasons (see Fig. 1–1 (c)). Firstly, the frequency of occurrence of text information in natural scene image is usually very low, and text information is always buried under all kinds of non-text outliers in cluttered background of natural scene. Thus background removal plays a significant role in text detection. Secondly, even though image regions containing text characters are detected from complex background, current optical character recognition (OCR) systems do not work well on the recognition of scene text, because they are mostly designed for scan documents. More effective feature representations and more robust models are required to improve the performance of scene text recognition. Thirdly, background textures, such as grid, window, and brick, even resemble text characters and strings. Although these challenging factors exist in the recognition of faces, human bodies and cars, many state-of-the-art algorithms [**18**] [**88**] have demonstrated effectiveness on those applications, because face and car, have relatively stable features. For example, a frontal face normally contains a mouth, a nose, two eyes, and two brows as prior knowledge. However, it is difficult to model the structure of text characters in scene images due to the lack of discriminative pixel-level appearance and structure features from non-text background outliers. Fourthly, unlike the text in scan documents, scene text usually appears in multiple colors, fonts, sizes and orientations. In addition, text information is probably destructed by motion blur or light exposure, or sometimes attached to curved planes.

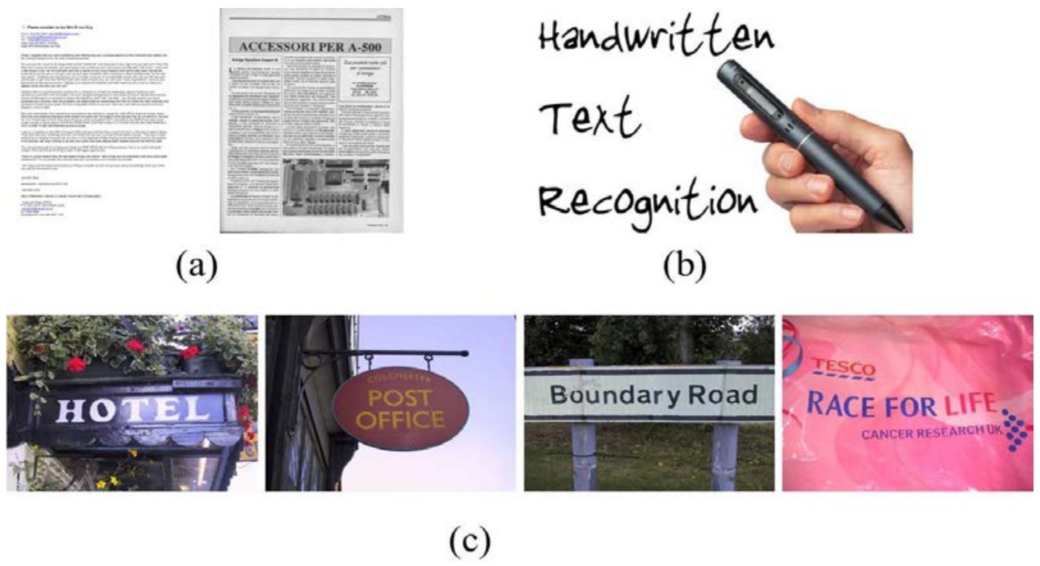


Fig. 1–1 Some examples of (a) scan document text; (b) handwritten text; (c) scene text.

1.2. Baseline Solutions

To overcome the above challenges, we need to solve two problems, 1) how to model text layout and structure so that it can be distinguished from non-text background outliers, which is scene text region detection; 2) how to model the structure of scene text characters so that the category of a given character can be correctly predicted, which is scene text character prediction.

To solve the two problems as mentioned above, our framework of scene text extraction is divided into two functional modules in the baseline solutions [**104**] [**36**], which are scene text detection and scene text recognition. The two modules generate text detector and text recognizer respectively, and they have evolved into two research topics. Scene text detection is to localize the image regions containing text and strings and filter out most background interferences. Some detection methods also segment the text strings in detected text regions into independent characters for recognition. Scene text recognition is to transform image-based text strings in detected text regions into readable ASCII codes. Fig. 1–2 illustrates the flowchart of the proposed framework.

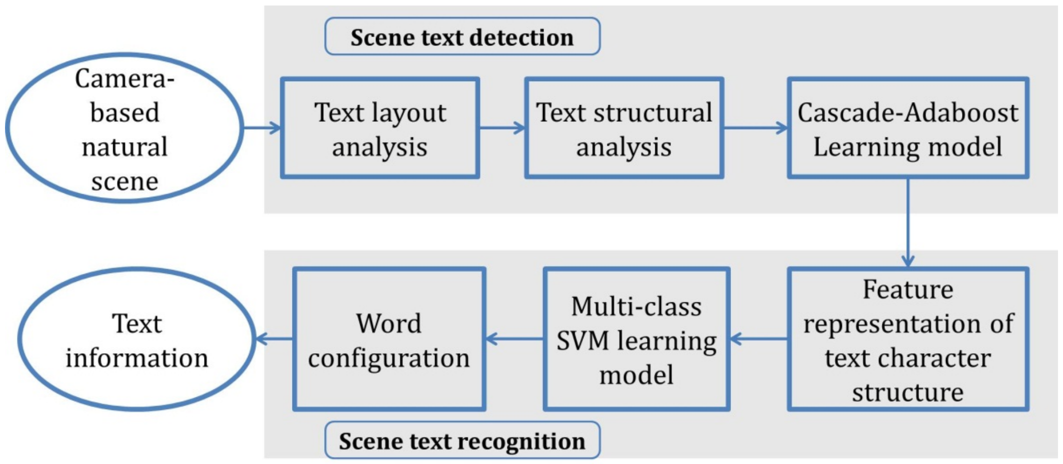

Scene text detection
Camera-based natural scene
Text layout analysis
Text structural analysis
Cascade-Adaboost Learning model
Feature representation of text character structure
Multi-class SVM learning model
Word configuration
Text information
Scene text recognition

Fig. 1–2. The flowchart of our proposed scene text extraction framework.

In the following chapters, we present the solutions of these problems. In Chapter 2, we present the methods of extracting connected components of possible text characters. Then the connected components of candidate text characters are grouped into fragments of possible text strings by layout analysis in Chapter 3. Chapter 2 and Chapter 3 present layout analysis, which models specific extrovert appearances of scene text. Chapter 4 describes the structural feature representation of text string fragment for filtering out false positive text strings obtained by layout analysis. In Chapter 5, we start scene text recognition in the detected text regions, by designing feature representation of scene text character for text character classification. Chapter 4 and Chapter 5 present structure analysis, which models specific introvert structure of scene text. Chapter 6 describes word-level recognition on the basis of STC prediction and conditional random field (CRF) model. Chapter 7 and Chapter 8 present the experimental results and discussions on scene text detection and scene text recognition respectively. From Chapter 9 to Chapter 12, we present 4 promising applications that involve scene text extraction. We conclude our studies and present our future work in Chapter 13.

Chapter 2 SCENE TEXT CHARACTER COMPONENTS

2.1. PROBLEM FORMULATION

It is difficult to confirm the existence of text information in scene images with complex background because the occurrence frequency of scene text is usually very low. Generally, the camera captured scene text characters or strings only occupy small image regions, which are surrounded by a large number of non-text background outliers.

To extract text from natural scene with complex background, we start with the search of semantic objects in natural scene, and then pick out the candidate regions containing scene text characters which satisfy the predefined geometrical constraints. They are defined as candidate character components, and represented by connected components or bounding boxes.

In most cases, scene text character appears in the form of connected component in uniform color in contrast to its surrounding attachment surface, and scene text character brings in high edge density and specific gradient distribution in contrast to the dominant plain regions in a scene image. Based on these two layout characteristics of scene text, we propose two methods in this chapter, extracting candidate character components from scene image, which are color-based partition in Section 2.3 and gradient-based partition in Section 2.4. The two methods can generally handle low-quality capture or illumination variation in camera-based scene image, obtaining clear and complete text characters from cluttered background.

2.2. PREVIOUS WORK

Some previous publications focused on scan document text. To extract text information from camera-captured document images (i.e. most part of the captured image contains well organized text with clean background), many algorithms and commercial optical character recognition (OCR) systems have been developed [**5**] [**76**]. Liang *et al.* [**45**] used texture flow analysis to perform geometric rectification of the planar and curved documents. Burns *et al.* [**7**] performed topic-based partition of document image to distinguish text, white spaces and figures. Banerjee *et al.* [**3**] employed the consistency of text characters in different sections to restore document images from severe degradation based on the model of Markov Random Field. Lu *et al.* [**52**] proposed a word shape coding scheme through three topological features of characters for text recognition in document image. All the above algorithms share the same assumption that locations of text characters are approximately predictable and background interference does not resemble text characters.

Some previous publications focused on the extraction of text character components from cluttered background of natural scene image. They can be roughly classified into two categories. The first category focuses on text region initialization and extension by using distinct features of text characters. To extract candidates of text regions, Kasar *et al.* [**39**] first assigned a bounding box to the boundary of each candidate character in the edge image and then detected text characters based on the boundary model (i.e. no more than 2 inner holes in each bounding box of alphabets and letters in English); Tran *et al.* [**84**] calculated ridge points in different scales to describe text skeletons at the level of higher resolution and text orientations at the level of low resolution; Liu *et al.* [**47**] designed a stroke filter to extract the stroke-like structures; Sobottka *et al.* [**78**]combined a top-bottom analysis based on color variations in each row and column with a bottom-top analysis based on region growing by color similarity; Hasan *et al.* [**30**] and Park *et al.* [**67**] designed robust morphological processing; Wolf *et al*. [**95**] improved Otsu's method [**65**] to binarize text regions from background, followed by a sequence of morphological processing to reduce noise and correct classification errors. To group together text characters and filter out false positives, these algorithms employed similar constraints involved in character, such as the minimum and maximum size, aspect ratio, contrast between character strokes and background, the number of inner holes. However they usually fail to remove the background noise resulted from foliage, pane, bar or other background objects that resemble text characters. To reduce background noise, the algorithms in the second category partition images to blocks and then groups the blocks verified by the features of text characters. Lefevre *et al.* [**43**] further designed a fusion strategy to combine detectors of color, texture, contour, and temporal invariance respectively. In Weinman *et al.* [**92**] used a group of filters to analyze texture features in each block and joint texture distributions between adjacent blocks by using conditional random field. One limitation of these algorithms is that they used non-content-based image partition to divide the image spatially into blocks of equal size before grouping is performed. Non-content-based image partition is very likely to break up text characters or text strings into fragments which fail to satisfy the texture constraints. Thus some algorithms use the property stroke width consistency to find out possible text characters. In [**40**], local color quantization was performed to extract text from background, but it required that text and background had explicit color difference. In [**22**] [**21**], stroke width is calculated by using horizontal scan line to record the intensity variations around the edge pixels, usually a pair of impulses on the strokes with equal magnitudes and opposite directions. To avoid the limitations of horizontal scan of stroke width, Epshtein *et al.* [**22**] designed a stroke width transform to extract text characters with stable stroke widths. In addition, the color uniformity of text characters in natural scene image is taken into account for content-based partition [**13**] [**53**] [**61**] [**79**]. However the unexpected background noises might share the same colors with text characters, so texture features of characters are still required. The algorithms in our proposed framework belong to this category of partition and grouping, but our content-based partition is involved in both gradient features and color features.

2.3. Color-Based Partition

2.3.1. Absolute Color Reduction

A text string is mostly composed of character members in similar colors. To separate text strings from attachment surfaces and non-text outliers in different colors, we partition a scene image into several color layers by grouping pixels in similar colors into the same layer. We adopt a color reduction method based on color histogram and mean-shift clustering.

Firstly, image pixels located within a plane region are extracted to make statistics of dominant colors, while the pixels close to the border of two neighboring plane regions are removed because they appear in transitional colors rather than dominant colors. To achieve this goal, we employ Canny edge detector to obtain the edge map of a scene image. The pixels whose 3×3 neighborhoods do not contain any edge pixels in the edge map are selected to calculate color histogram of the scene image. Next, we map all the non-edge pixels of a scene image from spatial domain into RGB color space, and generate a color histogram of the scene image, as shown in Fig. 2–1.

Secondly, we need to find out high-density clusters from the color histogram and transform these clusters into representative color layers of the scene image. Thus mean-shift clustering algorithm is designed to cluster the pixels in similar colors in RGB space. K color centers $\left\{cl_i^{(0)} \middle| 1 \leq i \leq K\right\}$ are randomly initialized from the non-edge pixels, and each of them is assigned a cube neighborhood $CUBE_i$ with size l in RGB space, as shown in Fig. 2–1. Next, the color center of each cube is iteratively shifted to higher-density clusters in RGB space. In a color cube $CUBE_i$, each color value is weighted by its Euclidean distance from color center cl_i , and we calculate the weighted average of all color values in the i-th cube by Eq. (2–1).

$$cl_i^{(t+1)} = \frac{\sum_{p \in CUBE_i^{(t)}} cl_p \cdot \left|cl_p - cl_i^{(t)}\right|}{\sum_{p \in CUBE_i^{(t)}} \left|cl_p - cl_i^{(t)}\right|} \qquad (2\text{–}1)$$

where $cl_i^{(t)}$ represents the i-th color center at the t-th iterative round. Then $CUBE_i$ is shifted to higher-density color cluster by using the weighted average $cl_i^{(t+1)}$ as new center and keeping its size l. The distance between the weighted average $cl_i^{(t+1)}$ and the original center $cl_i^{(t)}$ is defined as shifting distance. The cube shifting is iteratively performed until the shifting distance is smaller than a predefined threshold.

Thirdly, if several color centers obtained from mean-shift algorithm are close enough to each other, they are further merged together and we use their mean value as a final color center. The final color centers correspond to a color layer as shown in Fig. 2–1.

The number of color layers of a scene image depends on the number of initial color centers K and the setting of cube size l. In most cases, the final number of color layers is smaller than K because of the merge process. The smaller the cube size is, the fewer pixels are covered in a cube, which results in the shorter shifting distance. It is more difficult for the cubes to shift close to each other to complete a merge. Thus the total number of color layers will increase, and many objects in scene image will be broken up into small parts in different color layers. However, the cube size cannot be too large, so that the pixels far from each other in color space are still grouped into the same color layer. In that case, color-based partition fails to extract candidate character components. In our experiments, we set $K = 12$ and $l = 64$.

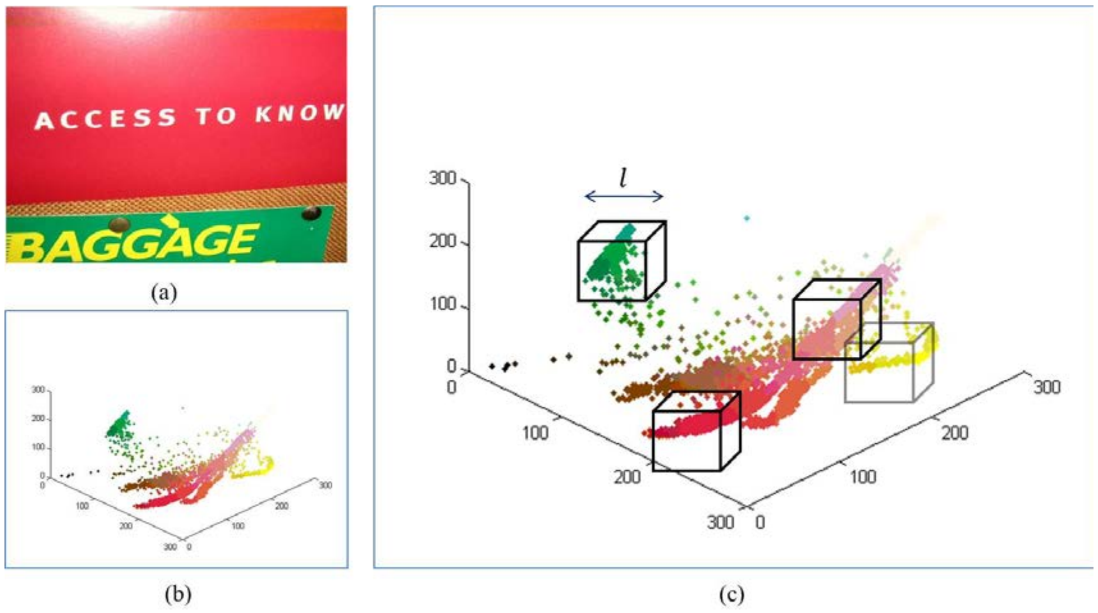


Fig. 2–1 (a) A scene image with multiple colors; (b) color distribution in RGB space; (c) four initial cube color clusters with radius h.

Some examples of the color-based image partition method are displayed in Fig. 2–2. Each input image is partitioned to several color layers. A color layer consists of only one foreground color on white background. It can be regarded as a binary map of candidate character components. Then connected component analysis is performed to label foreground regions of connected pixels, which is then extracted as candidate character components.

The candidate character components not satisfying the pre-defined geometrical constraints are removed from each color layer. In color-based partition, we follow the geometrical constraints of size and aspect ratio as defined in gradient-based partition. Section 2.6 presents the detailed description of the geometrical constraints.

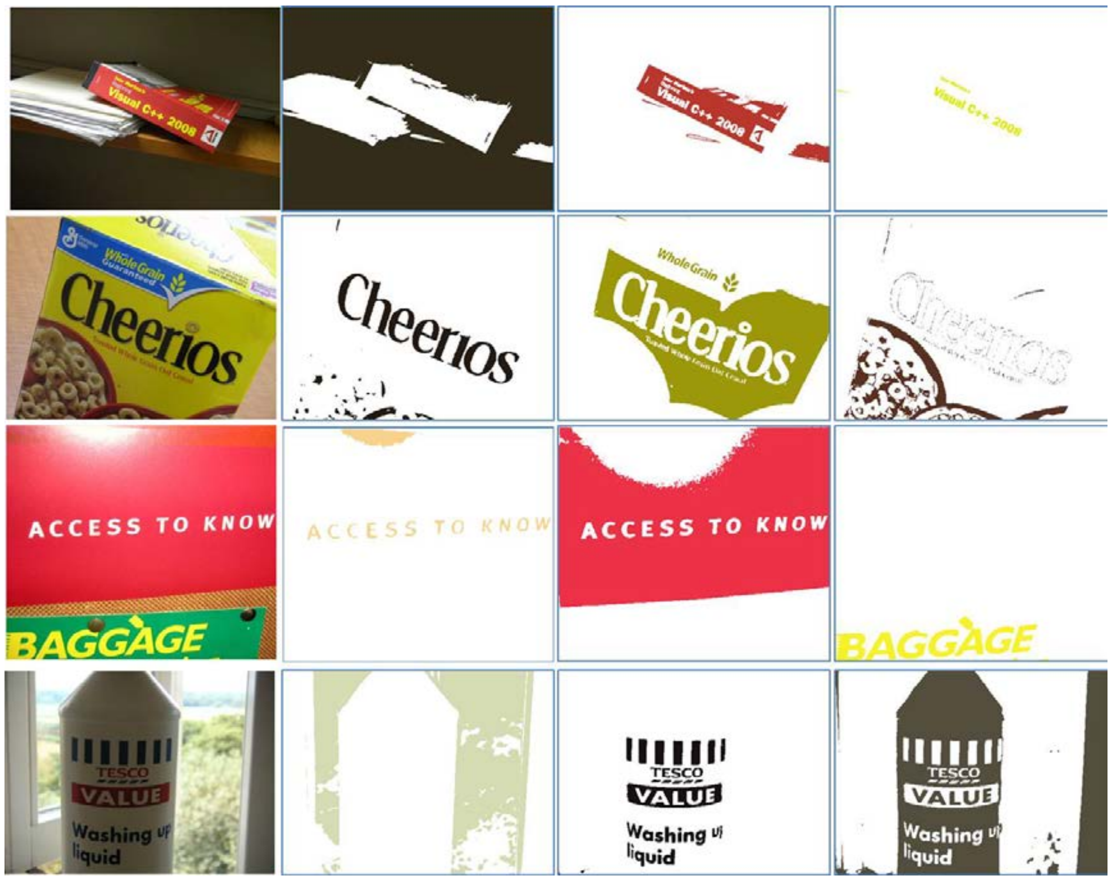


Fig. 2–2 Some examples of color-based partition, where the left column contains original images and other columns contain the corresponding dominant color layers.

2.3.2. Bigram Color Reduction

Camera-based scene images usually have complex background filled with non-text objects in multiple shapes and colors. In these images, text strokes, characters, and strings keep conspicuous by consistent colors. Thus many color-based clustering methods of text localization and text segmentation are designed in [**13**] [**53**] [**61**]. However, these clustering methods ignore the color differences among neighboring pixels around the object boundaries. Compared with absolute color value that describes only a planar region, color difference value covers the color information of two neighboring planar regions. Color difference is more suitable for the analysis of object shape and texture because it models local texture more accurately and handles illumination changes more robustly [**46**].

We observe that text information is generally attached to a plane carrier as attachment surface. The attachment surface consists of pixels with uniform color near the character boundaries but outside the character strokes, as shown in Fig. 2–3. We define the color uniformity of both text and attachment surface as bigram color uniformity, modeled by a color-pair composed of their colors. For text and attachment surface, the color-pair reflects their respective color uniformity as well as color difference between them. Text boundaries on the border of text and its attachment surface are described by characteristic color-pairs, and we are able to extract text by distinguishing boundaries of characters and strings from those of background outliers based on color-pairs (see Fig. 2–3(c)). Here, we design a method of grouping object boundaries with similar color–pairs into respective maps, which are called boundary layers. The contours in each boundary layer are extracted to be candidate character components. The next section presents detailed descriptions of the clustering method.

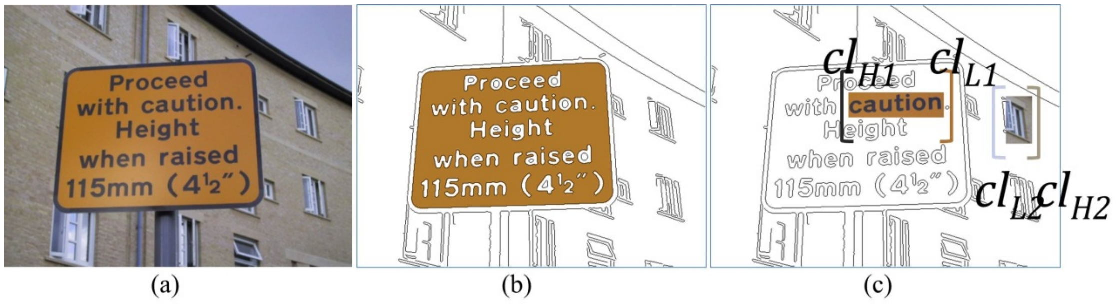


Fig. 2–3 (a) A scene image. (b) Attachment surface of text. (c) Two examples of color-pairs corresponding to the boundaries at the signage board and window grid respectively.

To reduce mutual interferences between text strings, text boundaries in different positions should be assigned into different boundary layers as possible, even though they have uniform color values and similar color differences. In other words, spatial position of object boundaries can be used as additional features in color-based partition. But most previous color-based clustering methods did not take into account the spatial positions of text. One reason is that additional spatial information makes the color statistics of all pixels time-consuming. In our proposed clustering method, bigram color uniformity and spatial-based partition schemes are combined to partition the edge map of a scene image into several boundary layers. Since only edge pixels at object boundaries are involved in the clustering process, the computational time consumption is largely reduced.

In natural scene images, an initial map of object boundary is calculated from Canny edge detection [**11**]. Edge pixels at boundaries are obtained from either large neighboring color differences that are greater than a threshold of Canny detector or the 8-neighborhood connection to an existing edge pixel. We describe the edge pixels by characteristic color-pairs. In $n \times n$ neighborhood Nb_n of an edge pixel P_e, we find out the two pixels with maximum color difference among all pairs of pixels. Their color values are used as observation of the color-pair across two sides of the boundary where the edge pixel is located. We denote the color with lower intensity component by $\boldsymbol{cl_L}$ and the other one by $\boldsymbol{cl_H}$ (see Fig. 2–3(c)). In RGB space, color values $\boldsymbol{cl_L}$ and $\boldsymbol{cl_H}$ both have three dimensions. If the boundary belongs to a text character or string, the $\boldsymbol{cl_L}$ and $\boldsymbol{cl_H}$ represent colors of text and attachment surface respectively. Moreover, the coordinates sp_x and sp_y of the edge pixel P_e are used as observation of spatial positions. Then an observation vector x of the edge pixel can be defined by cascading the color values of the two pixels with maximum color difference in neighborhood and the spatial coordinates of the central edge pixel. To normalize the dimensions of color-pair and spatial position observation, the coordinates sp_x and sp_y are extended into three dimensions as $\boldsymbol{sp_x} = \{sp_x, sp_x, sp_x\}$ and $\boldsymbol{sp_y} = \{sp_y, sp_y, sp_y\}$. Thus the edge pixel

P_e is described by an observation vector $x = [\boldsymbol{cl_L}, \boldsymbol{cl_H}, \boldsymbol{sp_x}, \boldsymbol{sp_y}]$, which is a 12-dimesnional point in observation space.

To extract text boundaries from scene images, we cluster the observation points of edge pixels into several groups such that edge pixels with similar color-pairs and spatial positions are assigned into identical boundary layer. In this process, GMM is employed to analyze the distributions of observation points of edge pixels. At first, K-means clustering is applied to calculate K centers of observation points, which are used as initial means $\mu_i\ (1 \leq i \leq k)$ of the Gaussian mixture distributions. Then the corresponding K variances $\sigma_i\ (1 \leq i \leq k)$ are calculated from the means of observation points. Thus we can initialize a group of Gaussian distributions. By labeling each of them with a weight, the expectation of GMM is represented by Eq. (2–2).

$$P(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{i=1}^{K} w_i p_i(x|\mu_i, \sigma_i) \qquad (2\text{–}2)$$

where $\boldsymbol{x}$ represents observation points, w_i represents the weights of the i-th Gaussian distribution in the mixture set, and μ_i and σ_i represent mean and variance of the i-th Gaussian distribution. Next, over the observation points of edge pixels, EM algorithm is applied to obtain maximum likelihood estimate of the GMM parameters, including weights, means, and variances of the K Gaussian distributions. In EM process, the GMM parameters are iteratively updated by Eq. (2–3) from their initial values derived by K-means clustering. In this equation, N is the number of observation points and t denotes the t-th iteration. This iterative update is performed until the log likelihood $\log \prod_{j=1}^{N} P(x_j|\boldsymbol{\mu}, \boldsymbol{\sigma})$ is convergent.

$$w_i^{t+1} = \frac{1}{N}\sum_{j=1}^{N} p_i\left(x_j \middle| \mu_i^t, \sigma_i^t\right)$$

$$\mu_i^{t+1} = \frac{\sum_{j=1}^{N} p_i\left(x_j \middle| \mu_i^t, \sigma_i^t\right) x_j}{\sum_{j=1}^{N} p_i\left(x_j \middle| \mu_i^t, \sigma_i^t\right)} \tag{2–3}$$

$$\sigma_i^{t+1} = \frac{\sum_{j=1}^{N} p_i\left(x_j \middle| \mu_i^t, \sigma_i^t\right)(x_j - \mu_i^{t+1})^2}{\sum_{j=1}^{N} p_i\left(x_j \middle| \mu_i^t, \sigma_i^t\right)}$$

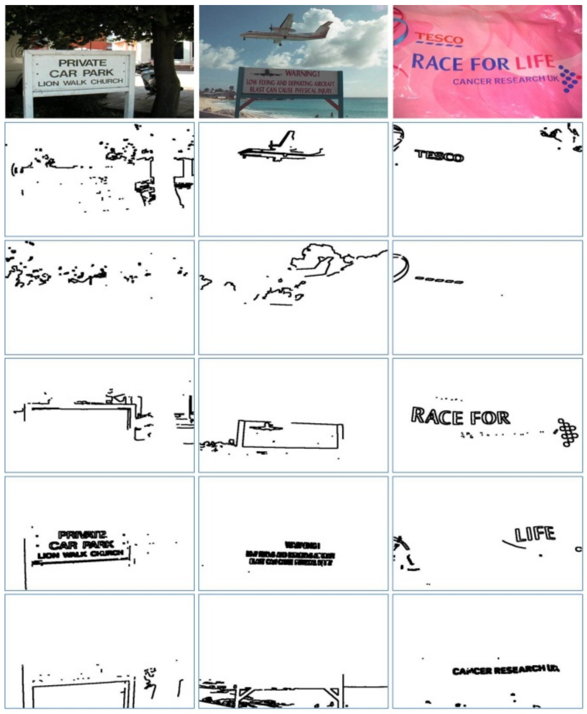
Then, boundary layer is built from each of the K Gaussian distributions under the parameters derived by EM. For an edge pixel, if it generates maximum likelihood in the i-th Gaussian distribution, it will be assigned into the i-th boundary layer B_i by Eq. (2–4) as follows.

$$\boldsymbol{X_i} = \left\{x_j \in \boldsymbol{x} \middle| \forall k \in [1, K], p_i\left(x_j \middle| \mu_i, \sigma_i\right) \geq p_k\left(x_j \middle| \mu_k, \sigma_k\right)\right\}$$

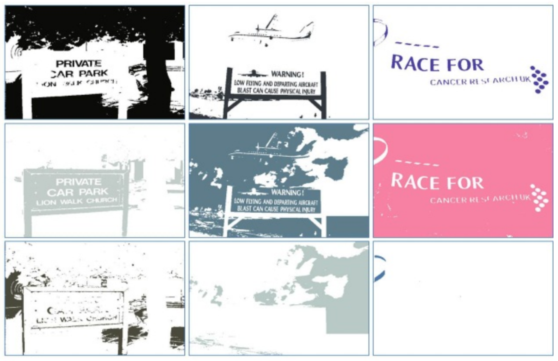
$$B_i(p) = \begin{cases} 1 & \text{if } x_p \in \boldsymbol{X_i} \\ 0 & otherwise \end{cases} \tag{2–4}$$

Furthermore, the expected value μ_i of the i-th Gaussian distribution provides a mean color-pair $\{\boldsymbol{cl_L^i}, \boldsymbol{cl_H^i}\}$ to label all edge pixels at the layer B_i.

Fig. 2–4(a) illustrates three examples of boundary layers after EM-based clustering. Fig. 2–4(b) presents the corresponding results of regular color reduction. As shown in the first two examples, color reduction fails to completely extract text from background outliers, leaving the boundaries of tree and plane on the boundary layer of text. It is because color reduction does not employ the spatial position difference between text and background outlier. In the third example, color reduction fails to extract the words "TESCO" and "LIFE", but fuses them into attachment surface in similar color. Color reduction quantizes the dominant colors through statistics of absolute color values, so neighboring objects with color difference lower than some threshold are very probably regarded as a complete object. However, our proposed clustering method quantizes the color-pairs around edge pixels instead of absolute color values at all pixels. A color-pair can be successfully extracted as long as it covers enough edge pixels to compose its boundary layer, even though the difference between the pair of colors is small.



(a)



(b)

Fig. 2–4. (a) Examples of boundary layers from scene images; edge pixels with similar color-pairs and spatial positions are grouped into the same layer. Boundaries at different veridical positions are assigned into different boundary layers because of y-coordinate spatial information in clustering process. (b) Results of color reduction based on clustering of absolute color values, where white region represents background, and color region consists of the pixels that are grouped to the layer.

Since most of the involved text strings in our experiments are approximately horizontal, the spatial positions of text boundaries can be estimated only in y-coordinates. Thus the dimension of an observation point is reduced to 9 as $x = [\boldsymbol{cl_L}, \boldsymbol{cl_H}, \boldsymbol{sp_y}]$. In our experiments on scene images, the number of Gaussian mixtures is $K = 5$, which generates the best results of boundary clustering in most natural scene images. If K is too small, text boundary cannot be effectively extracted from complex background. If K is too large, the boundary clustering process will lose the tolerance to color variation within a character or string. In that case, text boundary is probably broken into several fragments and assigned into different boundary layers.

By using bigram color uniformity and spatial coherence, edge map of scene image is partitioned into several boundary layers, in which the object contours are extracted to be candidate character components.

2.4. Gradient-Based Partition

2.4.1. Contour Search

Besides color uniformity, the existence of text characters and strings always generates specific gradient distribution and edge density variation. We develop several gradient-based partition methods to extract candidate character components from background.

A straightforward method is to extract the object contours from edge map of scene image, and then find out the possible boundaries of scene text characters from the object contours. Since the text characters have limited size and continuous contour, we could search for possible text characters by their boundaries in edge map by Canny detector [**11**]. A boundary is in the form of a set of connected edge pixels. Fig. 2–5 illustrates all available boundaries in a natural scene image. Each boundary is a possible text character, and we can define some geometrical constraints as presented in Section 2.6 to find out scene text characters.

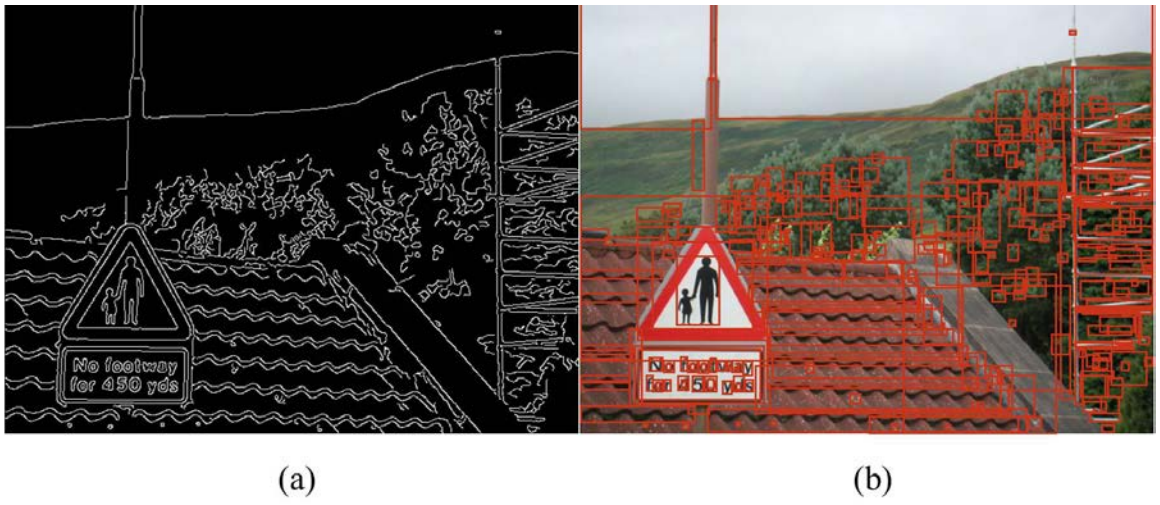


Fig. 2–5. (a) Canny edge map of a scene images. (b) Bounding boxes of all available edge boundaries in the form of connected edge pixels, obtained from edge map.

However, without predefined constraints like color uniformity as described in last section, the contours of text characters are mixed with the contours of background objects, and it is difficult to distinguish them. Thus we adopt gradient-based information to extract text from background.

2.4.2. Connecting Path

Object contours can only be modeled by some superficial geometrical constraints without taking account of gradient variation around character boundary. We propose a more robust method using the gradient distributions of scene text characters to separate candidate character components from scene image.

In our method, each pixel is mapped to connecting path of a pixel couple, defined by two edge pixels p and q on edge map with approximately equal gradient magnitudes and opposite directions, as shown in Fig. 2–6(a). Each pixel couple is connected by a path. Then the distribution of gradient magnitudes at pixels of the connecting path is computed to extract candidate character component.

Fig. 2–6(a) depicts that a character boundary consists of a number of pixel couples. We model the character by distribution of gradient magnitudes and stroke size including width, height and aspect ratio. The partitioned components are calculated from connecting path of pixel couple across the pixels with small gradient magnitudes.

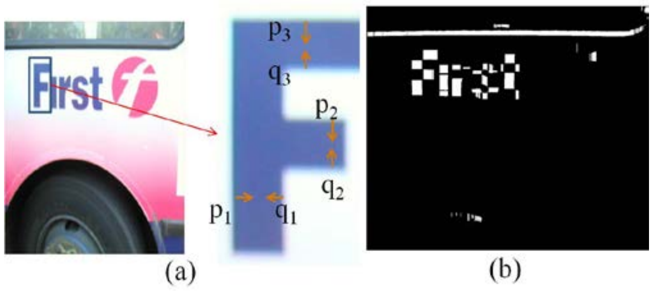


Fig. 2–6 (a) Examples of pixel couples; (b) connecting paths of all pixel couples are marked as white foreground while other pixels are marked as black background.

On the gradient map, $G_{mag}(p)$ and d_p $(-\pi < d_p \leq \pi)$ are used respectively to represent the gradient magnitude and direction at pixel p. We take an edge pixel p from edge map as starting point and probe its partner along a path in gradient direction. If another edge pixel q is reached where gradient magnitudes satisfy $|G_{mag}(p) - G_{mag}(q)| < 20$ and directions satisfy $\left|d_q - \left(d_p - (d_p/|d_p|) * \pi\right)\right| < \pi/6$, we obtain a pixel couple and its connecting path from p to q. This algorithm is applied to calculate connecting paths of all pixel couples. Fig. 2–6(b) marks all the connecting paths shorter than 30 as white foreground. To perform the gradient-based partition, we employ gradient magnitude at each pixel on the connecting path and length of connecting path l describing the size of connected component to be partitioned. The partition process is divided into two rounds. In the first round, the length range of connecting path is set as $0 < l \leq 30$ to describe stroke width. For each pixel couple whose connecting path falls on this length range, we establish an exponential distribution of gradient magnitudes of the pixels on its connecting path, denoted by Eq. (2–5)

$$g\left(G_{mag};\lambda\right)=\lambda\exp(-\lambda G_{mag}) \tag{2–5}$$

where the decay rate λ is estimated by $\hat{\lambda}=l/\sum G_{mag}$. A larger decay rate leads to faster falloff of gradient magnitudes on a connecting path. It means that the connecting path crosses a number of pixels with small gradient magnitudes on gradient map. This feature is consistent with the intensity uniformity inside the character strokes.

Thus we can set a threshold of the decay rate to extract candidate character components, and remove some background outliers. Ideally, the larger the threshold is, the better the results of character extraction are, as shown in Fig. 2–7. However, the decay rate only ensures the extraction of the text characters with large enough size and resolution. Thus in our experiments, we set a relatively small value 0.1 to extract as many candidate character components as possible, and leave the removal of false positives in the process of extracting scene text strings. To extract the complete stroke in rectangle shape, we start the second round to analyze the connecting paths along the stroke height (larger side). Since aspect ratio of rectangle stroke is no more than 6:1, we extend length range of connecting path to $0<l\leq 180$. Then we repeat the same analysis of gradient magnitudes for the connecting path not only falling on this length range but also passing through the regions of candidate character components obtained from the first round. At last, we perform morphological close and open as post processing to refine the extracted connected components, as shown in Fig. 2–8. The refined connected components are taken as candidate character components.

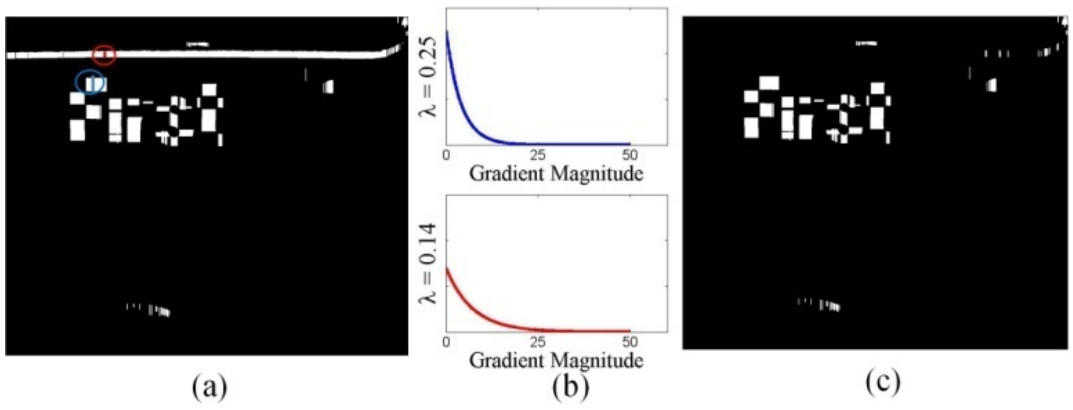


Fig. 2–7. (a) Two connecting paths of pixel couples marked by blue and red circles respectively. (b) The corresponding exponential distribution of gradient magnitudes on the connecting paths. (c) Partitioned components obtained from the first round.

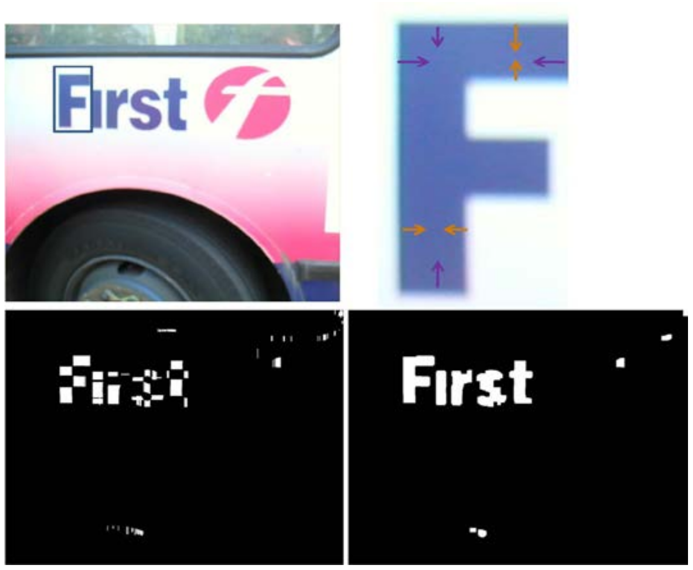


Fig. 2–8. Connecting path of pixel couple along the larger side of rectangle stroke is analyzed in the second round partition. Top row shows pixel couples in purple across the larger side of rectangle strokes. Bottom row presents the partitioned components obtained from the first round and the second round respectively.

The gradient-based partition generates a binary map of candidate character components on black background. By the model of local gradient features of character stroke, we can filter out background outliers while preserving the structure of text characters. Fig. 2–9 demonstrates that the gradient-based partition performs better on character component extraction than morphological processing.

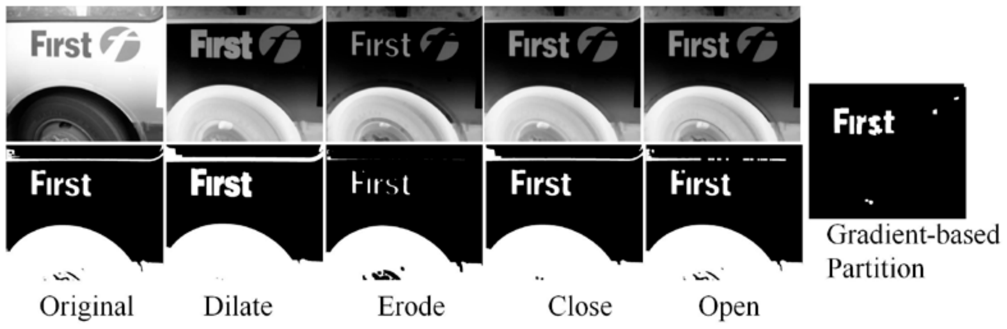


Fig. 2–9. Connected components obtained from direct morphological processing on gray images and corresponding binary images. We compare results of four morphological operators with result of our gradient-based partition.

2.5. Two Popular Methods of Extracting Text Character Components

In addition to gradient-based and color-based partition, many other methods were proposed to effectively extract text character components. Two popular methods are maximal stable extremal region (MSER) [**55**] [**99**] [**101**] [**59**] and stroke width transform (SWT) [**22**]. The two common use methods are able to generate clear and complete candidate character components. Thus we present their technical details and compare them with our proposed gradient-based and color-based partition methods in this section.

2.5.1. Maximal Stable Extremal Region

MSER detector was proposed in 2002 [**55**], which has been used as a blob detection technique for a long time in computer vision field. MSER is defined based on an extension of the definitions of image and set. Let image $I: D \subset \mathbb{Z}^2 \rightarrow S$, where D denotes the set of all pixels in the image, and S is a totally ordered set with reflexive, anti-symmetric and transitive properties. An adjacent relation is defined as A. Then an MSER region Q is defined as a contiguous subset of D, such that for each $p, q \in Q$, there is a sequence $p, a_1, a_2, a_3, \dots, a_n, q$ and $pAa_1, a_1Aa_2, \dots, a_iAa_{i+1}, \dots, a_nAq$. From the perspective of image pixel, the point p, q and a_i represent the coordinates of pixels, and the adjacent relation is mostly defined by intensity threshold, that is, for two neighboring pixel a_i and a_{i+1}, we have a_iAa_{i+1} if and only if $|I(a_{i+1}) - I(a_i)| \leq Threshold$, where $I(a_i)$ denote the intensity at pixel a_i. As shown in Fig. 2–10, MSER generates connected components of text characters in scene image, while edge map only gives the contours of text characters. Further, MSER map filters out the foliage thoroughly.

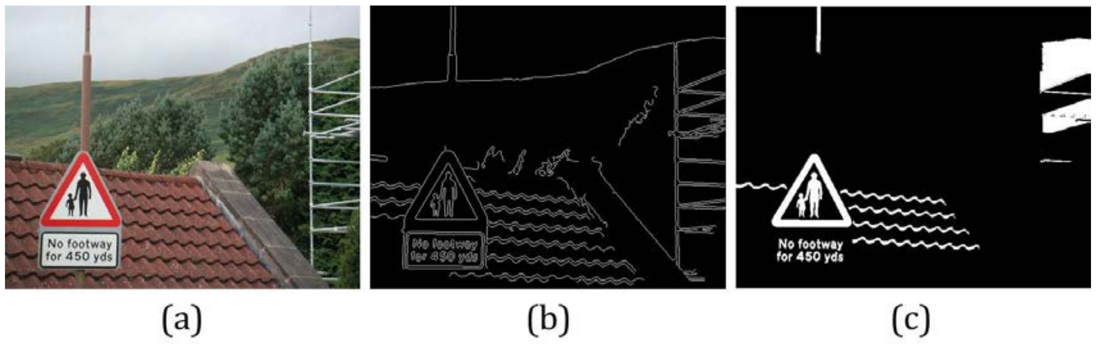


Fig. 2–10. (a) Original natural scene image. (b) Canny edge map. (c) Some MSER regions as white foreground.

Since MSER cannot confirm the intensity polarity of text and attachment surface, that is, not able to distinguish white-text-in-black-background from black-text-in-white-background, both text and attachment surface will be extracted as candidate character components in MSER map. However, attachment surface components can be easily removed by defining some geometrical properties.

Moreover, MSER has several specific properties compatible with the requirement of extracting candidate character components from natural scene image. Firstly, MSER is invariant to affine transformation of image intensities. Secondly, MSER extraction is very stable since a region is selected only if its support is nearly the same over a range of thresholds. Thirdly, MSER is scale-invariant, that is, it is able to extract candidate character components in multiple scales without any pre-processes of original natural scene image. Fourthly, MSER extraction is very efficient, because its time complexity in the worst case is $\mathrm{O}(n)$ where n represents the number of pixels in the image.

2.5.2. Stroke Width Transform

SWT was proposed in 2010 [**22**], in which a local operator associated with stroke width is designed to model the specific structure of text character and extract text character components from non-text background outliers.

Stroke serves as a basic unit to compose a text character. Stroke is defined as a contiguous part of an image that forms a band of nearly constant width, as the region with red boundary shown in Fig. 2–11. Stroke width is calculated as the distance between two pixels with similar gradient magnitude and opposite gradient directions. SWT labels the pixels located inside the torso of a stroke by its width, and transforms a natural scene image into a stroke width map.

The implementation of SWT is as follows. Firstly, canny edge detector was applied to obtain the edge map of a natural scene image. Secondly, at each edge pixel p in the edge map, gradient direction d_p is calculated, which is approximately perpendicular to the stroke orientation. Thirdly, a pixel ray is generated from edge pixel p along the gradient direction d_p for raster probe, until the arrival of another edge pixel q, whose gradient direction is d_q. If the two gradients have approximately equal magnitudes and opposite directions (see Fig. 2–11), all the pixels inside the segment path ended at p and q are labeled by the length $|p-q|$. In [**22**], a constraint is defined in Eq. (2–6) to check whether gradient magnitude is approximately identical and gradient direction is approximately opposite.

$$\|d_p - d_q\| \leq \frac{\pi}{6} \tag{2–6}$$

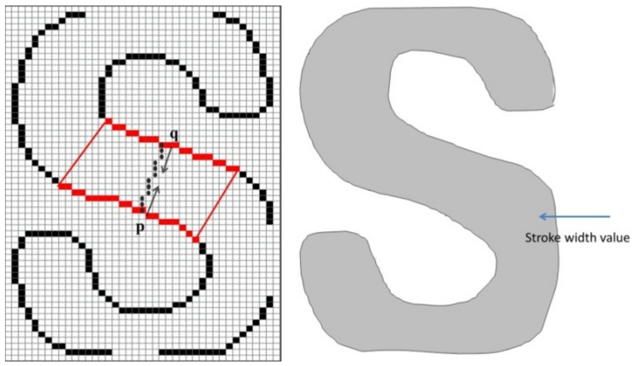


Fig. 2–11. Stroke width transform: A character stroke is represented by the red boundary. The two pixels p and q are on the boundary of a character stroke. Stroke width is calculated as the distance between two pixels which have similar gradient magnitude and opposite gradient direction. Each pixel is assigned a value of stroke width to be a stroke width map, and the pixels with similar values in this map are grouped together to compose the connected components on the right, which will be used as candidate character component in later processes.

If a pixel is labeled more than once, the minimum length value will be assigned to it. Then pixels labeled by similar stroke width values are grouped as candidate connected components of text characters. As shown in Fig. 2–11, the pixels within the stroke of character “S” will have similar values in stroke width map, and we can adopt breadth first search algorithm to search for similar pixels and group them into candidate character components. SWT can be used to extract inner structure feature of text configuration, which will be presented in Section 4.3.

2.5.3. Comparisons with Gradient-based and Color-based Methods

Here, we briefly summarize the gradient-based partition and color-based partition methods. Both of them play a significant role in the extraction of candidate character components from natural scene image. Based on gradient and color, a group of text layout and structural features can be designed to separate text information from background outliers. However, they also have several differences. First, color-based model describes the torso of a text character while gradient-based model mostly describes the boundary of a text character. In ideal case, character torso and boundary can be transformed into each other, but this transformation usually fails in cluttered background. The torso model can better tolerate low-resolution than boundary model, so color-based methods usually give better performance than gradient-based method. Secondly, color-based model requires the statistics of the whole scene image to obtain proper parameters of color clustering, while the gradient-based model requires only local computing of gradient information. From this point, gradient-based methods usually have higher efficiency than color-based methods. Therefore, we need to seek a proper solution to combine the advantages of the two methods in real applications.

2.6. Geometrical Constraints of Candidate Character Components

Text layout is modeled to remove most background outliers in natural scene image. It includes color uniformity, gradient distribution, and character alignment, which are extrovert characteristics of text.

To remove the non-text background outliers from the set of candidate character components, we define a group of geometrical constraints. In these constraints, a candidate character component C is described by several geometrical properties: $height(\cdot)$,$width(\cdot)$, $coorX(\cdot)$, coor$Y(\cdot)$, $area(\cdot)$, and $numInner(\cdot)$, which represent height, width, centroid x-coordinates, centroid y-coordinates, area, and the number of inner candidate character components respectively.

We define a group of geometrical constraints based on above measurements to ensure that the preserved candidate character components are truly text characters as possible. Since we will further use character grouping and text structure modeling to remove false positive candidate character components, the constraints defined in this step are not very strict.

$$\begin{gathered}
height(C) > 15\,pixels \\
0.3 \leq \frac{width(C)}{height(C)} \leq 1.5 \\
numInner(C) \leq 4 \\
\frac{1}{10} \cdot ImageWidth \leq coorX(\cdot) \leq \frac{9}{10} \cdot ImageWidth \\
\frac{1}{10} \cdot ImageHeight \leq coorY(\cdot) \leq \frac{9}{10} \cdot ImageHeight
\end{gathered} \tag{2–7}$$

The involved geometrical constraints are presented in Eq. (2–7). First of all, the candidate character component cannot be too small, or else we will regard it as background noise. It also means that our whole framework of scene text extraction requires enough resolution of camera-captured scene text information. Second, the aspect ratio of a true positive character should be located in a reasonable range. Under a threshold of aspect ratio, we might also remove some special text characters like “l”, but it is very possible to restore this false removal by generating text string fragments as described in Chapter 3. Third, we define some constraints related to the number of nested candidate character components as presented in [**39**]. Fourth, we observe that many background outliers obtained from the above partition methods are located at the margins of camera-based scene image. Thus the candidate character components whose centroids are located at the 1/10 margin of the camera-based images will not be taken in account in further processes.

Compared with the geometrical constraints defined in [**59**] [**99**], the geometrical constraints we defined are very straightforward and easy to implement, and most of them are just external geometrical measurements of a candidate character component, such as size and aspect ratio, without association with inner structure of text character (except the number of holes, but we set much weak threshold than that in [**39**]). Also, different from the methods in [**59**] [**99**], we do not use any learning model to decide the parameters. Instead, all the involved geometrical constraints in this step are weak conditions, with the preservation of true positive text characters in higher priority than the removal of false positive background outliers. Therefore, only the obvious background outliers are filtered by the geometrical constraints. The remaining false positive candidate character components will be handled in the extraction of text string fragments.

2.7. Summary of Contributions

In this chapter, our main contributions are two methods of extracting candidate character components. The first method is based on the decomposition of edge map into multiple layers according to bigram color uniformity, and the second method is based on the searching of edge pixel couple according to gradient distribution. Both these methods can effectively extract possible text characters in the form of connected components from natural scene image with complex background.

Chapter 3 SCENE TEXT STRING FRAGMENTS

3.1. PROBLEM FORMULATION

The image partition presented in last chapter creates a set of candidate character components in the form of connected components or boundary contours $\mathbb{C} = \{C_i | 1 \leq i \leq |\mathbb{C}|\}$ from an input image, where $|\mathbb{C}|$ represent the total number of candidate character components from image partition. Most candidate character components in $\mathbb{C}$ are not true positive scene text characters but non-text background objects in uniform color or some portions of an object under uneven illumination. Geometrical constraints of single candidate character components cannot remove them, so we design more discriminative layout characteristics of scene text from high-level perspective. Text in natural scene mostly appears in the form of words and phrases, but not single characters. It is because words and phrases are more informative text information, while single character usually serves as a sign or symbol. Words and phrases are defined as text strings, and we attempt to find out possible text strings by combining neighboring candidate character components. In this chapter, a combination of neighboring candidate character components is defined as text string fragments. Since we have not confirmed the true positive characters, a text string fragment is further named as *candidate string fragment*. It can be regarded as a portion of text string itself, or a portion of text region containing text string.

In this chapter, assuming that a text string consists of at least two character members in alignment, we propose two methods to extract candidate string fragments from candidate character components, which are adjacent character grouping in Section 3.3 and text line fitting in Section 3.4 respectively. In this chapter, we will continue the use of the geometrical properties of candidate character components and their corresponding symbols defined in Section 2.6. In addition, we define $D(C_1, C_2)$ as the distance between the centroids of two candidate character components C_1 and C_2, and generate a group of constraints associated with scene text string fragments in Section 3.5.

3.2. Previous Work

Many researches on the extraction of text string fragment have been previously published. Phan *et al.* [**69**] performed line-by-line scan in edge images to combine rows and columns with high density of edge pixels into text regions. Gao *et al.* [**26**] and Suen *et al.* [**80**] performed heuristic grouping and layout analysis to cluster edges of objects with similar color, position and size into text regions. However these algorithms are not compatible with slanted text lines. Myers *et al.* [26] rectified the text line in 3D scene images by using horizontal and vertical features of text strings, but their work does not focus on detecting text line on complex background. Akoum *et al.* [**2**] employed gradient-based analysis to localize and recognize car license plates. Ma *et al.* [**56**] and Zhang *et al.* [**103**] designed multi-scale edge features detect scene text, and edge-based features are also adopted in our framework to model text-specific structure. In [**22**] [**99**] [**101**] , a group of geometrical constraints was defined to link neighboring candidate character components that probably belonged to the same text string. However, most of these algorithms just set the parameters of geometrical constraints by subjective assignment. The optimized parameters were chosen from performance evaluation of text region detection in a benchmark dataset. It means that these parameters might not be generalized to real environments with different text patterns from the ones in the dataset. Yao *et al.* [**99**] combined self-designed geometrical properties of text patches into Random Forest learning model, which automatically set optimized parameters of geometrical properties and generated an effective text classifier to extract true position text strings. Ze *et al*. [**100**] proposed a method of cylinder object unwarping to extract text information from non-planar surfaces.

3.3. Adjacent Character Grouping

Text strings in natural scene images usually appear in horizontal alignment, namely, each character in a text string has at least one sibling at adjacent positions. A text character and its siblings have similar sizes and proper distances. The candidate character components corresponding to non-text outliers can be removed if they do not have any siblings.

Now the main problem is how to decide whether two candidate character components C_1 and C_2 can be regarded as siblings of each other. According to our observations and statistical analysis of text strings, we define 4 geometrical constraints as follows.

1) Considering the approximate horizontal alignment of text strings in most cases, the centroid of candidate character component C_1 should be located between the upper-bound and lower-bound of the other candidate character component C_2, that is, $coorY(C_1) \geq coorY(C_2) - height(C_2) * T_1$ and $coorY(C_1) \leq coorY(C_2) + height(C_2) * T_1$

2) Two adjacent characters should not be too far from each other despite the variations of width, so the distance between two connected components should not be greater than T_2 times the width of the wider one, that is, $|coorX(C_1) - coorX(C_2)| \leq T_2 \cdot max\big(width(C_1), width(C_2)\big)$

3) For text strings aligned approximately horizontally, the difference between y-coordinates of the connected component centroids should not be greater than T_3 times the height of the higher one, that is, $|coorY(C_1) - coorY(C_2)| \leq T_3 \cdot max\big(height(C_1), height(C_2)\big)$.

4) If the connected components are obtained from gradient-based partition as described in Section 2.4, the color difference between them should be lower than a predefined threshold T_4 because the characters in the same string have similar colors, that is, $|\boldsymbol{cl}(C_1) - \boldsymbol{cl}(C_2)| \leq T_3$.

In our system, we set $T_1 = 0.5, T_2 = 3, T_3 = 0.5$ and $T_4 = 40$. For each candidate character component C_i, a sibling set $S(C_i)$ is generated, where $1 \leq i \leq |\mathbb{C}|$ and $|\mathbb{C}|$ is the number of candidate character components obtained from image partition. First, an empty sibling set is initialized as $S(C_i) = \emptyset$. We transverse all the candidate character components expect C_i itself. If a candidate character component C_i' satisfies all above constraints with C_i, we add it into the sibling set as $S(C_i) := S(C_i) \cup \{C_i'\}$. Second, all the sibling sets compose a set of adjacent groups $\Lambda = \{A_i | A_i := S(C_i)\}$, where a sibling set is initialized to be adjacent group A.

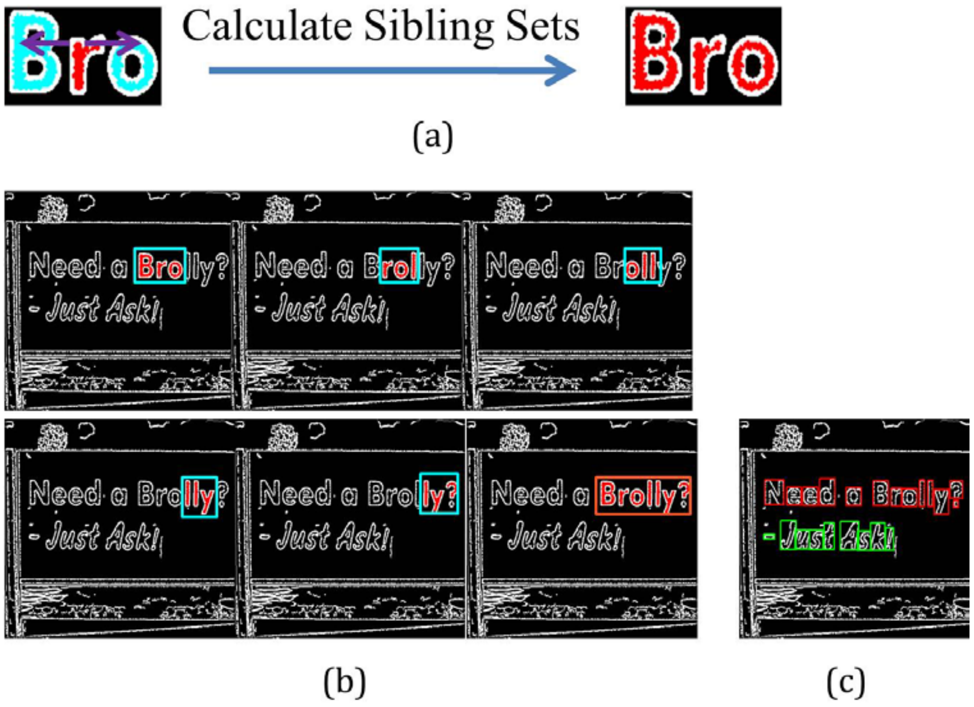


Fig. 3–1. (a) Sibling group of the connected component 'r' where 'B' comes from the left sibling set and 'o' comes from the right sibling set; (b) Merge the sibling groups into an adjacent character group corresponding to the text string "Brolly?"; (c) Two detected adjacent character groups marked in red and green respectively.

Third, the set of adjacent groups is iteratively updated by merging the overlapping adjacent groups. An adjacent group is a group of candidate character components that are probably character members of a text string. As Eq. (3–1), if two adjacent groups A_i and A_j in Λ have intersection, they will be merged into one adjacent group. This merging operation is iteratively repeated until no overlapping adjacent groups exist.

$$\forall A_i, A_j \in \Lambda, if\ A_i \cap A_j \neq \emptyset, then\ A_i \coloneqq A_i \cup A_j\ and\ A_j \coloneqq \emptyset \tag{3–1}$$

We summarize our method of adjacent character grouping in Table 3–1 in detail. In the resulting set of adjacent groups, each adjacent group A_i is a set of candidate character components in approximate horizontal alignment, which will be regarded as a candidate string fragment, as shown in Fig. 3–1.

The above method of adjacent character grouping assumes horizontal alignment of scene text string. However, it can be extended to text strings in arbitrary orientations by just modifying the geometrical constraints and merging method as the following 3 steps.

1) The above constraint 1 ensures the horizontal alignment of sibling candidate character components, and we can remove it to accept sibling candidate character components in arbitrary orientations.

2) The above constraints 2 and 3 ensures proper distances of two sibling candidate character components in horizontal and vertical orientation. To extend them into arbitrary orientations, we combine them into Eq. (3–2) and Eq. (3–3).

$$D(C_1, C_2) \coloneqq \sqrt{\left(coorX(C_1) - coorX(C_2)\right)^2 + \left(coorY(C_1) - coorY(C_2)\right)^2}$$

$$D(C_1, C_2) \leq T_5 \cdot \frac{max(width(C_1), width(C_2\ ))}{\sin\theta} \tag{3–2}$$

$$\theta \coloneqq \arctan\left\{\frac{coorY(C_1) - coorY(C_2)}{coorX(C_1) - coorX(C_2)}\right\} \qquad if\ coorX(C_1) \neq coorX(C_2)$$
$$\theta \coloneqq \pi/2 \qquad if\ coorX(C_1) == coorX(C_2) \tag{3–3}$$

3) An adjacent group is a set of collinear candidate character components, and they are now in arbitrary orientations, not have to be horizontal. Thus we add one more attribute to describe an adjacent group, which is orientation. In the process of merging adjacent characters A_i and A_j, besides $\mathrm{A}_i \cap \mathrm{A}_j \neq \emptyset$ we should add one more condition on the consistence of their orientations $\left|\mathrm{ori}(\mathrm{A}_i) - \mathrm{ori}(\mathrm{A}_j)\right| \leq \pi/6$. In the merging process as $\mathrm{A}_i \coloneqq \mathrm{A}_i \cup \mathrm{A}_j$ in Table 3–1, we will also update the orientation of A_i by fitting the line through all the candidate character components in A_i and A_j.

By the above 3 modifications, the method of adjacent character grouping is able to extract candidate string fragments in arbitrary orientations. However, without the assumption of horizontal alignment, the number of ways of combining the candidate character components largely increase, therefore more false positive string fragments will be generated.

Table 3–1. The procedure of adjacent character grouping

```
Input: The set of candidate character components $\mathbb{C} = \{C_i | 1 \leq i \leq |\mathbb{C}|\}$
Output: The set of adjacent groups $\Lambda$

for each pair of candidate character components $C_i, C_i' \in \mathbb{C}$
    if $coorY(C_i) \geq coorY(C_i') - height(C_i') * T_1$ and $coorY(C_i) \leq coorY(C_i') + height(C_i') * T_1$
      & $D(\text{coor}X(C_i),, \text{coor}X(C_i')) \leq T_2 * \max\{width(C_i), width(C_i')\}$
      & $D(\text{coor}Y(C_i), \text{coor}Y(C_i')) \leq T_3 * \max\{height(C_i), height(C_i')\}$
      & difference of mean RGB color value is less than $T_4$
        $S(C_i) := S(C_i) \cup C_i'$
        $S(C_i') := S(C_i') \cup C_i$
    endif
endfor

for each candidate character component $C_i \in \mathbb{C}$
    $A_i := S(C_i)$;
    $\Lambda := \Lambda \cup \{A_i\}$;
Endfor

while there exists overlapping adjacent groups $\mathrm{A}_i$ and $\mathrm{A}_j$ such that $\mathrm{A}_i \cap \mathrm{A}_j \neq \emptyset$
    for each pair of overlapping adjacent groups $\mathrm{A}_i, \mathrm{A}_j \in \Lambda$
        $\mathrm{A}_i \coloneqq \mathrm{A}_i \cup \mathrm{A}_j$;
        $\mathrm{A}_j := \emptyset$;
    endfor
endwhile

We obtain a set of adjacent groups $\Lambda$ after the iterative merge
Filter out false positives by the three filters decided by geometrical properties.
Calculate image regions of candidate string fragments based on the adjacent groups.
```

3.4. Text Line Fitting

Many applications require extracting candidate string fragments in non-horizontal orientations. The adjacent character grouping method can deal with non-horizontal text strings as long as their orientations are within -15^o to 15^o. To handle text strings in arbitrary orientations, we develop another method of text line fitting to directly combine the candidate character components in linear alignment. First the centroid of each candidate character component is calculated, and then their collinear centroids are cascaded into text lines. The set of centroids of candidate character components is denoted as M in Eq. (3–4). Now the problem is how to find out the subsets of collinear centroids

$$M = \{m | m = \langle coorX(C), coorY(C) \rangle\} \tag{3–4}$$

$$L = \left\{ G \middle| \begin{array}{c} G \subseteq M, |G| \geq 3, \forall m_i, m_j, m_k \in G, \\ they\ are\ character\ centroids, \\ and\ they\ are\ colinear. \end{array} \right\} \tag{3–5}$$

where C denotes a candidate character components obtained from image partition, M denotes the set of centroids, and L denotes the set of text lines which are composed of collinear centroids.

A naive solution is to search for satisfied centroid groups in the power set of M, but the complexity of this algorithm will be $O(2^{|M|})$ where $|M|$ represents the number of centroids in the set M. We design an efficient algorithm to extract regions containing text strings. Firstly we check the collinearity for each group of three centoirds m_i, m_j and m_k in the set M as Eq. (3–5). We calculate the length difference Δd and incline angle difference $\Delta\theta$ for the two line segments $m_i m_j$ and $m_j m_k$, as shown in Fig. 3–2. The three centroids are approximately collinear if $1/T_6 \leq \Delta d \leq T_6$ and $\Delta\theta \leq T_7$. In our system, we set $T_6 = 2$ and $T_7 = \pi/12$. Thus they compose a preliminary fitted line $l_u = \{m_i, m_j, m_k\}$ where u is the index of the fitted line. Secondly, after finding out all the preliminary fitted lines, we apply Hough transform to describe the fitted line l_u by $\langle r_u, \theta_u \rangle$, resulting in $l_u = \{m | h(r_u, \theta_u, m) = 0\}$ where $h(r_u, \theta_u, m) = 0$ is equation of the fitted line in the Hough Space. Thirdly, we check each centroid to see whether it is located around the fitted line within a predefined error. If yes, this centroid is added into the fitted line, and Hough equation of the fitted line is updated by re-calculating the collinearity in the presence of newly-added centroids. Fig. 3–2 illustrates an example of text line fitting, in which both text string and non-text background noises are extracted in the form of text string fragments as long as they satisfy spatial collinearity. Table 3–2 summarizes our algorithms in detail.

$$\Delta d = \frac{D(m_i, m_j)}{D(m_j, m_k)} \tag{3 - 6}$$

$$\Delta\theta = \begin{cases} |\theta_{ij} - \theta_{jk}|, & if\ |\theta_{ij} - \theta_{jk}| \leq \frac{\pi}{2} \\ |\theta_{ij} - \pi - \theta_{jk}|, & if\ |\theta_{ij} - \theta_{jk}| > \frac{\pi}{2}\ and\ \theta_{ij} > \theta_{jk} \end{cases} \tag{3 - 7}$$

where d is length of line segment and $d > 0$, θ is the angle of incline and $0 \leq \theta < \pi$.

The centroids from false positive character components can also be aligned as a line. To remove these false positive fitted lines, several constraints are further defined to distinguish the fitted lines corresponding to text strings from those generated by non-text background outliers. Section 3.5 gives more explanations.

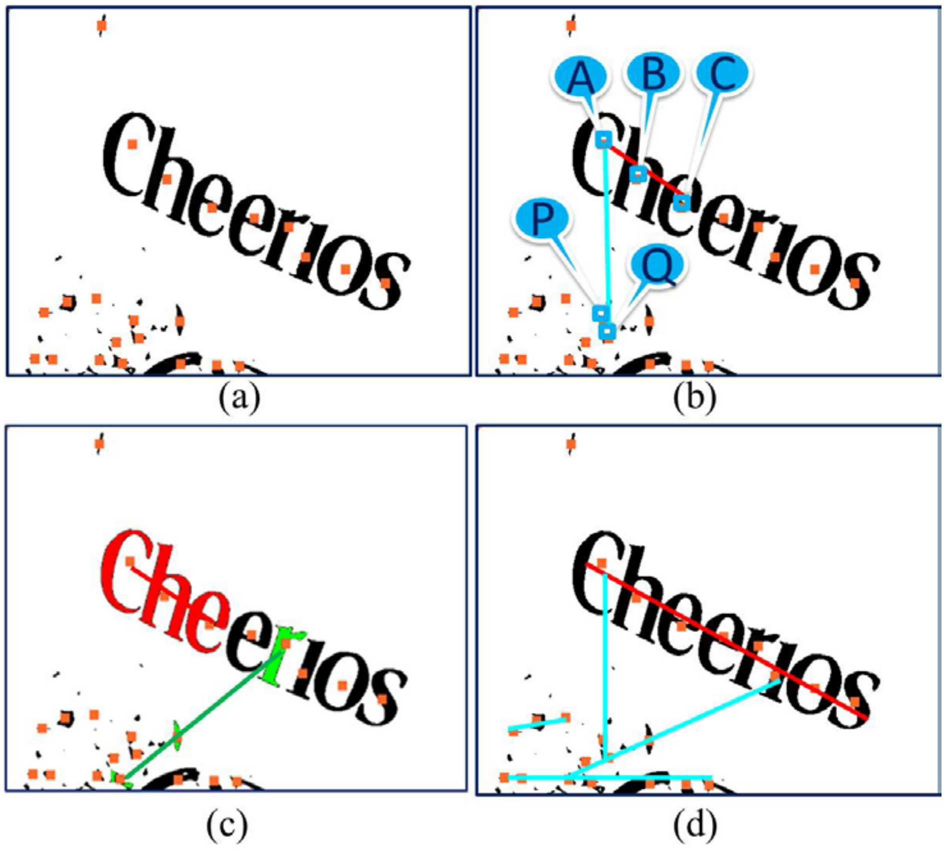


Fig. 3–2. (a) Centroids of connected components in a color layer; (b) $D(m_A, m_B)$ approximately equals to $D(m_B, m_C)$ in text region while $D(m_A, m_P)$ is much larger than $D(m_P, m_Q)$ in background, where $D(.)$ represents Euclidean distance; (c) Three neighboring connected components in red share similar areas while those in green have very different areas; (d) Resulting fitted lines from centroids cascading. Red line corresponds to text region while cyan lines are false positives to be removed.

Table 3–2. The procedure of text line fitting

```
Input: The set of candidate character components $\mathbb{C} = \{C_i | 1 \leq i \leq |\mathbb{C}|\}$
Output: The set of adjacent groups $L$

for every group of three points $m_i, m_j, m_k \in M$,
    calculate $\Delta d$ and $\Delta\theta$
    if $0.5 \leq \Delta d \leq 2$ and $\Delta\theta \leq \frac{\pi}{12}$
        $l_u := \{m_i, m_j, m_k\}$;
    endif
endfor

for every preliminary fitted line
    for every $m_t \in M$ and $m_t \notin l_u$
    $\langle r_u, \theta_u \rangle := Hough(l_u)$ where $l_u = \{m | h(r_u, \theta_u, m) = 0\}$
        if $h(r_u, \theta_u, m_t) < \varepsilon$
          & fitted line $l_u \cup \{m_t\}$ meets the two constraints
            $l_u := l_u \cup \{m_t\}$
            $\langle r_u, \theta_u \rangle := Hough(l_u)$ // re-calculating the Hough equation after adding in $m_t$
        endif
    $L := L \cup \{l_u\}$
    endfor
endfor

Filter out false positive fitted lines in $L$ by geometrical properties, and calculate extracted regions based on the positive fitted lines.
```

3.5. Geometrical Constraints of Candidate String Fragments

To remove the false positive adjacent groups and text lines that are generated by background noises, two constraints are further defined according to structure features of text strings.

In a text string, character members should have similar sizes. In adjacent character grouping, we measure this size similarity by $height(\cdot)$. In text line fitting, we only measure the directional alignment of candidate character components but not explicitly add the constraint of size similarity. Thus we define a geometrical constraint by the coefficient of variation of connected component areas to ensure the size similarity of character members in a text string.

In a text string, character members should have similar distances to their direct neighboring siblings. In the processes of both adjacent character grouping and text line fitting, we set the thresholds of neighboring distances as a constraint to confirm the valid combinations of adjacent groups and text lines. However, these constraints only provide a local measurement of distance similarity, without evaluating the geometrical structure of the whole string fragment. It is very possible that character members gradually increase neighboring distances, that is, the second distance is larger than the first neighboring distance within the predefined threshold, and the third neighboring distance is larger than the second neighboring distance within the predefined threshold, but the third one is so larger than the first one that their ratio exceeds predefined threshold. Thus we define a geometrical constraint by the coefficient of variation of neighboring distances of candidate character components to ensure the distance similarity in the whole text string.

The calculation of coefficient of variation (CV) for a candidate string fragment is presented in Eq. (3 – 8)

$$
\begin{aligned}
CVarea &= \frac{std(\boldsymbol{area}(\cdot))}{mean(\boldsymbol{area}(\cdot))} \\
CVdist &= \frac{std(\boldsymbol{dist}(.))}{mean(\boldsymbol{dist}(.))}
\end{aligned}
\qquad (3-8)
$$

where $\boldsymbol{area}(.)$ represents the vector of areas of all character members, $\boldsymbol{dist}(.)$ represents the vector of neighboring distances among all character members, $std(.)$ represents the standard deviation, and $mean(.)$ represents the mean value. The larger the $CVarea$ is, the larger the area dissimilarity is. The larger the $CVdist$ is, the larger the neighboring distance dissimilarity is. In our framework, the thresholds of $CVarea$ and $CVdist$ are set as 0.5 to restrict the dissimilarity.

3.6. SUMMARY OF CONTRIBUTIONS

In this chapter, our main contributions are two methods of extracting candidate string fragments, which are adjacent character grouping and text line fitting. The first method combines sibling candidate character components in similar size and horizontal alignment. The second method cascades the candidate character components in arbitrary orientations. Both these two methods can effectively extract possible text strings in alignment among the candidate character components.

Chapter 4 STRUCTURE MODELING FOR PREDICTING TRUE POSITIVE TEXT STRING FRAGMENTS

Text structure refers the inner configuration of text characters and strings, which are introvert characteristics of text. In this chapter, we model text structure to distinguish true positive text string from non-text background outliers among the candidate string fragments obtained from above steps.

4.1. PROBLEM FORMULATION

In Chapter 3, pixel-based text layouts are modeled to extract candidate character components and candidate string fragments from cluttered background. However, this possible scene text is generated from observations of layout characteristics, and statistic-based layout parameters are defined to model these layout characteristics. For example, according to the analysis and statistic of text regions cropped from scene image, we define that aspect ratio of text characters should not exceed 5 and stroke width should be no larger than 50. But these estimates do not effectively model the structural insight of characters and strings. They fail to filter out the background objects that are also composed of strokes in uniform color and consistent width, such as bricks, window grids, foliage and some objects rendered by specific illumination.

In this chapter, a candidate string fragment obtained by layout analysis is defined as a *fragment sample* in the form of image patch, as shown in Fig. 4–1. We will propose feature representations to model structural insights of fragment samples in Section 4.3. Each fragment sample is projected into a point in feature space, and cascaded-Adaboost learning model is then employed in Section 4.4 to train a robust classifier to pick true positive text out of the fragment samples obtained by layout analysis. As shown in Fig. 4–2, we design a learning-based algorithm for automatic localization of text regions in image.

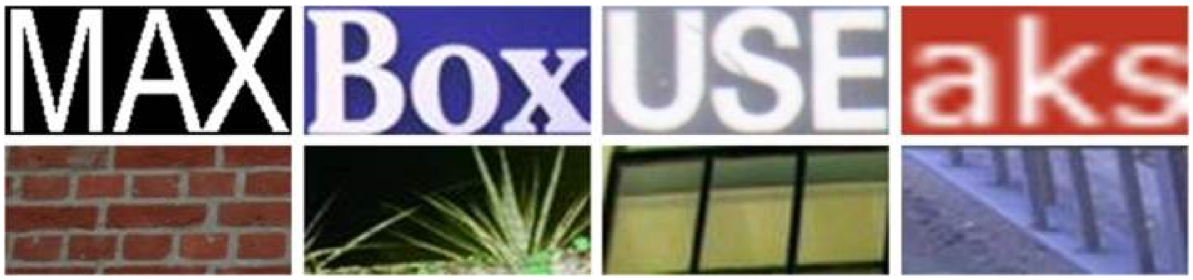


Fig. 4–1. Some examples of text fragment samples in the form of image patches.

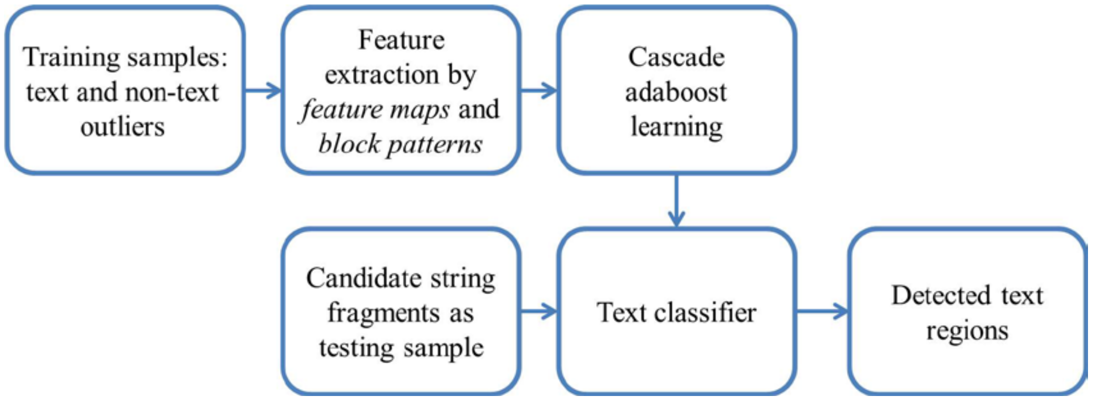

Training samples: text and non-text outliers
Feature extraction by *feature maps* and *block patterns*
Cascade adaboost learning
Candidate string fragments as testing sample
Text classifier
Detected text regions

Fig. 4–2. Diagram of the proposed Adaboost learning based text region localization algorithm by using stroke orientations and edge distributions.

4.2. Previous Work

Many researches on the modeling of text structure from fragment sample have been previously published. They adopted various structural feature designs and machine learning models.

Chen *et al.* [**15**] applied block patterns to gradient based maps and histograms to train text classifier in Adaboost model [**25**]. Hanif *et al.* [**28**] extracted mean difference, standard deviation, and HOG features of text characters to generate text detector under a Complexity Adaboost model. In [**42**], the responses of globally matched wavelet filters from text regions are used as features to train text classifier based on Support Vector Machines (SVM) model and Fisher model. In [**34**], Gabor filter was used to segment text from documents. Pan *et al.* [**66**] used steerable Gabor filters to extract rotation-invariant features of multiple scripts. Shi *et al.* [**74**] adopted gradient based curvatures to perform structural analysis of handwritten digits under a Bayes discriminant model. Jung *et al.* [**37**] proposed an algorithm of text line refinement by analyzing SVM score of text regions. In [**102**], Gabor-based features were designed to model inner structure of text and classify string fragments.

4.3. Feature Representations for Text String based on Feature Maps

4.3.1. Haar-like Block Patterns

To extract structural information of text strings from fragment samples, Haar-like filters are designed in the form of block patterns, as shown in Fig. 4–3. Each block pattern consists of white regions and gray regions in specific ratio. It will be resized into the same size as a fragment sample, and used as a mask. Then specific calculation metrics are defined based on these block patterns for extracting structural features, which will be described in detail in Section 4.3.3.

A simple idea of feature extraction is to apply these block patterns directly to the fragment samples, and calculate Haar-like features from intensity values of the image patches. However, unlike face detection [**88**], the sole intensity values cannot completely represent structure of text strings.

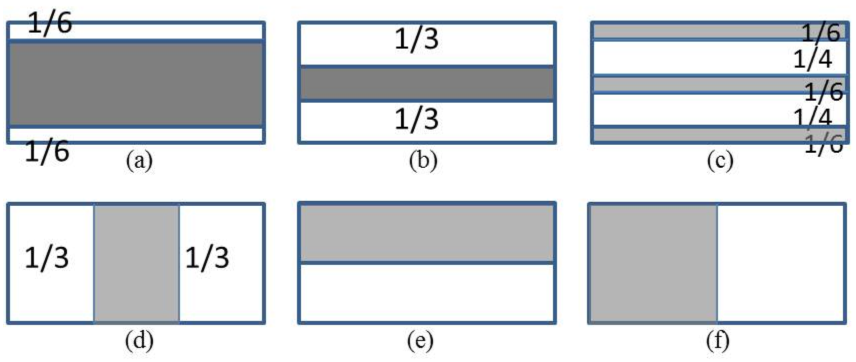


Fig. 4–3. Some examples of Haar-like block patterns to extract text structural features. Features are obtained by the absolute value of mean pixel values in white regions minus those in black regions.

4.3.2. Feature Maps

To model text structure, we design a set of feature maps for the fragment samples, in which the physical meaning of each pixel is transformed from intensity value to some measurements related to text structure. The structure-related measurements are mostly based on gradient, edge density and stroke. Here stroke is defined as a uniform region with bounded width and significant extent, which is a basic unit of text character.

In our framework, two novel feature maps are designed based on stroke orientation and edge density variation. Besides, gradient distribution and stroke width are also adopted as feature map, and they are calculated from Sobel operator and stroke width transform respectively. These feature maps are combined to build an Adaboost-based text classifier. The details of these feature maps are presented as follows.

A **Gradient**

The most intuitive text structural features originate from gradient magnitude and gradient orientation. As we know, text structure appears in the form of specific boundary, which is generated by stroke torso and attachment surface in uniform colors. The pixels around the text boundary have larger gradients than the pixels in plain regions. Thus the map of gradient magnitude and gradient orientation can be used to model text structure, as shown in Fig. 4–4.

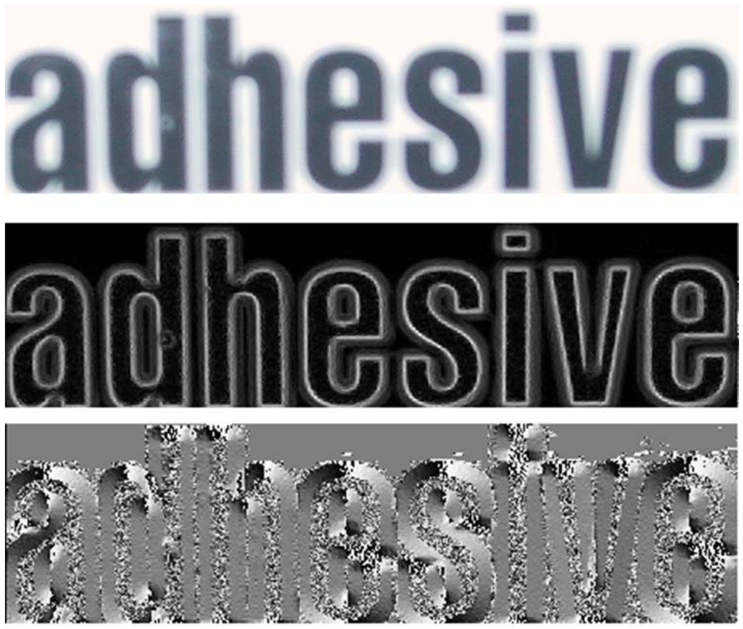


Fig. 4–4. From top to bottom, the figures represent the original text patch, the feature maps of gradient magnitude and the feature maps of gradient orientation respectively.

B **Text Stroke Width**

The stroke width transform (SWT) proposed in [**22**] is applied to the image patches to generate another feature map. The principle of SWT has been presented in Section 2.5.2. Some parameters of SWT algorithm are adjusted to be compatible with the fragment samples, since this algorithm is originally designed for the whole scene image. In SWT-based feature map, each pixel represents the width of its located stroke. If it is not located at the torso of text characters, we set it as 0.

C Text Stroke Orientation

Text characters consist of strokes with constant or variable orientation as the basic structure. Here, we propose a new type of feature, stroke orientation, to describe the local structure of text characters. From the pixel-level analysis, stroke orientation is perpendicular to the gradient orientations at pixels of stroke boundaries, as shown in Fig. 4–5. To model the text structure by stroke orientations in fragment sample, we propose a new operator to map a gradient feature of strokes to each pixel. It extends the local structure of a stroke boundary into its neighborhood by gradient of orientations. We use it to develop a feature map to analyze global structures of text characters.

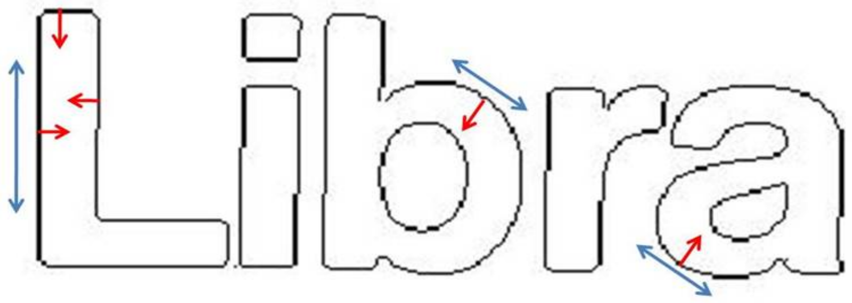


Fig. 4–5. A fragment sample of text showing relationships between stroke orientations and gradient orientations at pixels of stroke boundaries. Blue arrows denote the stroke orientations at the sections and red arrows denote the gradient orientations at pixels of stroke boundaries.

Given an image patch I, Sobel operators in horizontal and vertical derivatives are used to calculate 2 gradient maps $G_x$and $G_y$respectively. The synthesized gradient map is calculated as $G = \left({G_x}^2 + {G_y}^2\right)^{1/2}$. The Canny edge detector is applied on I

to calculate its binary edge map E. For a pixel p_0, we certify whether it is close to a character stroke by setting a circular range as $R(p_0) = \{p|d(p, p_0) \leq r\}$, where $d(.)$ denotes Euclidean distance, and r = 36 is the threshold of the circular range to search for edge pixels. We set this threshold because the text patches in our experiments are all normalized into height 48 pixels and width 96 pixels, and the stroke width of text characters in these normalized patches mostly does not exceed 36. If the distance is greater than 36, pixel p_0 would be located at background region far away from text character. In the range we select the edge pixel p_e with the minimum Euclidean distance from p_0. Then the pixel p_0 is labeled with gradient orientation at pixel p_e from gradient maps by Eq. (4 – 1).

$$\begin{gathered} p_e = \arg\min_{p \in P} d(p, p_0) \\ S(p_0) = \Upsilon\left(arctan\left(G_y(p_e), G_x(p_e)\right)\right) \end{gathered} \qquad (4 - 1)$$

where $P = \{p|p \in R(p_0),\ p \text{ is edge pixel}\}$. The stroke orientation calculated from $arctan$ will be in the range $(-\pi/2\,, \pi/2]$. To distinguish the pixels labeled with stroke orientation 0 and the unlabeled pixels also with value 0, Υ shifts the stroke orientations one period forward into the range $(3\pi/2\,, 5\pi/2]$, which removes the value 0 from the range of stroke orientations. A stroke orientation map $S(p)$ is output by assigning each pixel the gradient orientation at its nearest edge pixel, as shown in Fig. 4–6(a). The pixel values in stroke orientation map are then quantized into an N bin histogram in the range $(3\pi/2\,, 5\pi/2]$ (see Fig. 4–6(b)). In the feature extraction stage, strokes with identical or similar orientations are identified to describe the structure of text from one perspective. In the N bin histogram, we group the pixels at every b consecutive bins together to generate a multi-layer stroke orientation map, where strokes in different orientations are separated into different layers. Without considering the cyclic shifts of the bins, there are a total of $N - b + 1$ layers.

The range of stroke orientations $(3\pi/2, 5\pi/2]$ is quantized into $N = 16$ bins, so each bin corresponds to $\pi/16 = 11.25^{\circ}$ and $b = 3$ consecutive bins will cover a range of 33.75°. This span value is compatible with most character strokes in scene images, because the stroke orientations are always vertical, horizontal or approximate $30^{\circ}\sim40^{\circ}$ such as "W", "X", and arc components of "P", "D" *etc*. Since b is set to be 3 and N is set to be 16, each sample generates 14 layers of stroke orientation maps, where text structure is described as gradient features of stroke orientations. We can extract structural features of text from such stroke orientation maps.

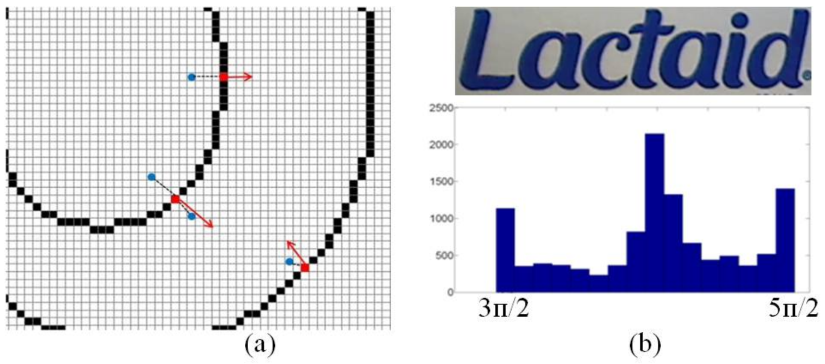


Fig. 4–6. (a) An example of stroke orientation label. The pixels denoted by blue points are assigned the gradient orientations (red arrows) at their nearest edge pixels, denoted by the red points. (b) A 210×54 fragment sample and its 16-bin histogram of quantized stroke orientations.

D Distribution of Edge Pixels

In an edge map, text characters appear in the form of stroke boundaries. The distribution of edge pixels in stroke boundaries also describes the characteristic structure of text. The most commonly used feature is edge density of text region. But the edge density measure does not give any spatial information of edge pixels. It is generally used for distinguishing text regions from relatively clean background regions. To model text structure by spatial distribution of edge pixels in fragment sample, we propose an operator to map each pixel of an image patch into the number of edge pixels in its cross neighborhood. At first, edge detection is performed to obtain an edge map, and the number of edge pixels in each row y and each column x is calculated as $N_R(y)$ and $N_C(x)$. Then each pixel is labeled with the product value of the number of edge pixels in its located row and in its located column. Then a 3×3 smooth operator w_n is applied to obtain the edge distribution feature map, as Eq. (4 – 2). In this feature map, pixel value reflects edge density in its located region, and the smoothed map better represents the discriminative inner structure of text characters.

$$D(x,y) = \sum_n w_n \cdot N_R(y_n) \cdot N_C(x_n) \qquad (4 - 2)$$

where (x_n, y_n) is neighboring pixel of (x,y) and w_n = 1/9 denotes the weight value.

4.3.3. Feature Value Generation

In feature map of a fragment sample, each pixel is transformed from intensity to some measurements related to text structure. Each pixel reflects text structural configuration from a local perspective. Several design schemes of feature maps have been presented in last section. By tuning the parameters of generating feature maps under a design scheme, we can obtain multiple feature maps. In our framework, we define 3 gradient maps, 2 stroke width maps, 14 stroke orientation maps, and 1 edge distribution map, to which 6 Haar-like block patterns are applied for calculating feature values.

Fig. 4–3 demonstrates the involved block patterns. In calculating a feature value for a given fragment sample in training set, we first generate one feature map of this fragment sample. Then we normalize one block pattern into the same size (height

48 pixels, width 96 pixels) as its feature map, and compute a feature value f by calculating the absolute difference between the sum/mean of pixel values in white regions and the sum/mean of pixel values in black regions. For the block patterns with more than 2 sub-regions (see Fig. 4–3(a-d)), the other metric of feature response is the absolute difference between the mean of pixel values in white regions and the mean of pixel values in black regions. Thus we obtain $6 + (6 - 2) = 10$ feature values for one feature map. The "integral image" algorithm is used in these calculations as in [**88**]. Since we define 20 feature maps, a fragment sample will generate a feature vector in $20 \times 10 = 200$ dimensions, as Eq. (4 – 3). Each dimension is defined as a structural element. We compute feature vectors for all the 51234 fragment samples in the training set. By using feature vector $\boldsymbol{f}^i$ of the *i*-th fragment sample as the *i*-th column**,** a feature matrix $\boldsymbol{F}$ is obtained by Eq. (4 – 4).

$$\boldsymbol{f}^i = \left[f_1^i, f_2^i, \dots, f_{200}^i\right]^T \tag{4 – 3}$$

$$\boldsymbol{F} = [\boldsymbol{f}^1, \boldsymbol{f}^2, \dots, \boldsymbol{f}^t, \dots, \boldsymbol{f}^{51234}] \tag{4 – 4}$$

The 200×51234 feature matrix is used for learning a text classifier in a Cascade-Adaboost model (see Section 4.4 for detail). A row of the feature matrix corresponds to a structural element, recording feature responses of a certain block pattern and a certain feature map on all fragment samples in training set.

4.4. CASCADED ADABOOST LEARNING

A supervised learning process is performed over a training set of fragment samples, in which text string fragments are labeled as positives and non-text outlier fragments are labeled as negatives. The learning process is to train a robust classifier H that is able to correctly predict whether a given fragment sample truly contains text or not.

4.4.1. Weak Classifiers

For each fragment sample s, a feature vector $\boldsymbol{f_s}$ in $M = 200$ dimensions is generated by 6 Haar-like block patterns, 2 metrics and 14 feature maps. Thus in a training set with $|N^+|$ positives and $|N^-|$ negatives, we can obtain a feature matrix in $(|N^+| + |N^-|) \times M$ dimensions. As mentioned above, each dimension corresponds to a structural element, and we define a pool of weak classifiers by setting split-position and split-polarity at each structural element.

In the process of Adaboost learning, weak classifier is defined as $\langle r, T_r, \rho \rangle$. The three parameters denote the r-th structural element (r-th row) $(1 \leq r \leq 200)$, a threshold of the structural element T_r, and polarity of the threshold $\rho \in \{-1,1\}$. The r-th structural element represents the r-th feature element. In each structural element r, linearly spaced threshold values are sampled in the domain of its feature values by Eq. (4 – 5).

$$T_r \in \left\{ T \middle| T = f_r^{min} + \frac{1}{N_T}\left(f_r^{max} - f_r^{min}\right)t \right\} \qquad (4 - 5)$$

where N_T represents the number of thresholds, f_r^{min} and f_r^{max} represent the minimum and maximum feature value of the r-th structural element, and t is an integer ranging from 1 to N_T. We set $N_T = 300$ in the learning process. Thus there are in total $200 \times 2 \times 300 = 120000$ weak classifiers denoted as $\mathcal{H}$. When a weak classifier $\langle r, \rho, T_r \rangle$ is applied to a sample with corresponding feature vector $\boldsymbol{f} = [f_1, \dots, f_r, \dots, f_{200}]^{\boldsymbol{T}}$, if $\rho f_r \geq \rho T_r$, it is classified as a positive, otherwise it is classified as a negative.

4.4.2. Inference

The Cascade-Adaboost classifier has proved to be an effective machine learning algorithm in real-time face detection [**88**]. The training process is divided into several stages. In each stage, a stage-specific Adaboost classifier is learned from a training set, which consists of all positives and the negatives incorrectly classified by previous Adaboost classifiers at this stage. We refer to this as a stage-Adaboost classifier in the following paragraphs. The Adaboost learning process at each stage is presented in detail in Appendix A.1.

All the stage-Adaboost classifiers are cascaded into the final Cascade-Adaboost classifier, as shown in Fig. 4–7. This model can robustly handle the imbalance between the number of positive samples and that of negative samples. The training process is divided into several stages. In the first stage, a portion of negative samples (with the same number as positive samples) is randomly selected. They are combined with all positive samples to train a stage-Adaboost classifier. Starting from the second stage, the positive samples keep the same, while only the negative samples that cannot be correctly classified by previous stage-Adaboost classifiers are adopted to train another stage-Adaboost classifier. The negative samples that have been correctly classified by previous stage-Adaboost classifiers will be discarded in later training process. In the whole training process, each stage-Adaboost classifier ensures that 99.5% of positive samples are correctly classified while 50% of negative samples are correctly classified. Thus a testing sample with positive ground-truth will have a $(0.995)^T$ probability of **correct** classification, and a testing sample with negative ground-truth will have $(0.5)^T$ probability of **incorrect** classification. where T represents the total number of stage-Adaboost classifiers.

When a testing fragment sample in the form of image patch is given, we first extract its feature vector based on the haar-like block patterns and feature maps, the same as a training fragment sample. Then the feature vector is input into the final classifier, it will classified as a text patch if all the cascaded stage-Adaboost classifiers determine it is a positive, and otherwise it will be classified as a non-text patch.

4.5. Summary of Contributions

In this chapter, our main contributions are designing feature maps to model text structure. The measurements related to text structure, including gradients, stroke width, stroke orientation, edge distribution, and wavelet response, are adopted to design feature maps of text/non-text patches. Each feature map is then developed into a weak classifier between text string fragments (positive samples) and non-text background outliers (negative samples). Then cascaded adaboost learning model is applied to train a text classifier to distinguish text from non-text patch. As one of our contributions, the cascaded learning model is adopted to solve text detection problem because of the imbalanced number of positive samples and negative samples in training dataset.

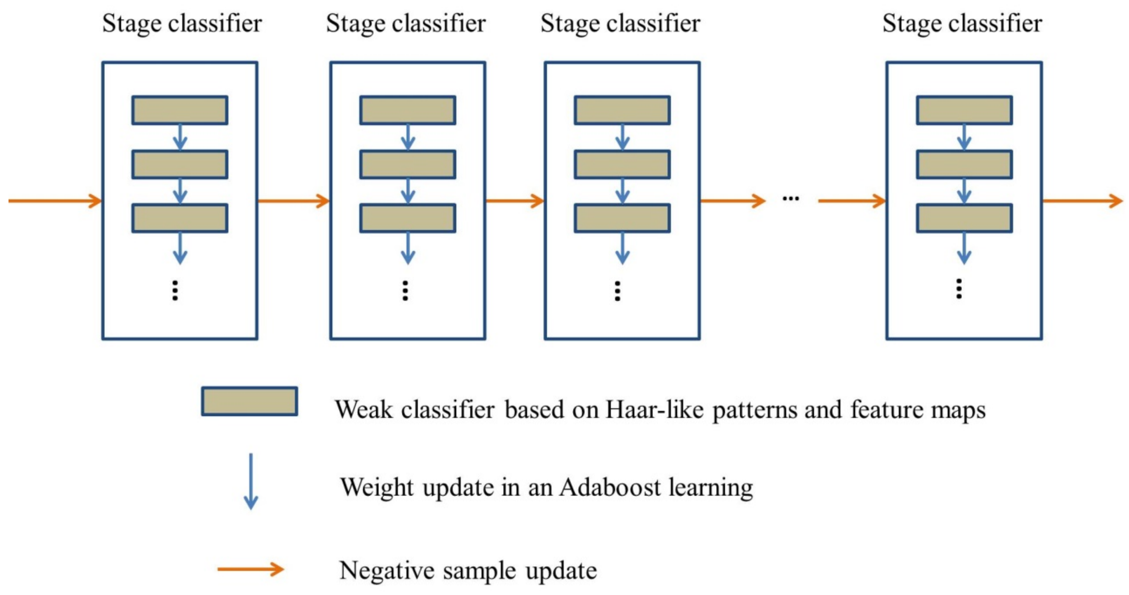

Stage classifier
Stage classifier
Stage classifier
Stage classifier
Weak classifier based on Haar-like patterns and feature maps
Weight update in an Adaboost learning
Negative sample update

Fig. 4–7. The flowchart of cascaded Adaboost classifier. It consists of a sequence of stage-Adaboost classifier. Each stage-Adaboost classifier combines an optimized set of weak classifiers. The set of negative fragment samples in training process keeps updated stage-by-stage, ensuring that a negative cannot be correctly classified by all previous stages.

Chapter 5 STRUCTURE MODELING FOR SCENE TEXT CHARACTER PREDICTION

5.1. PROBLEM FORMULATION

Scene text detection as presented in Chapter 3 and Chapter 4 is able to extract the image regions containing text information, and filter out most non-text background outliers. Structure modeling in last chapter is to distinguish text fragment sample from non-text background outlier. However, in this chapter, structure modeling is to distinguish the category of a text character in an image patch. It is used to transform the image-based text information in the detected regions to code-based text information. Scene text recognition is to generate readable text codes in the form of words and phrases from the detected text regions. Scene Text Character (STC) recognition, which generally includes feature representation to model character structure and multi-class classification to predict label and score of character class, mostly plays a significant role in scene text recognition. It serves as the basic process of word-level/phrase-level recognition.

Most off-the-shelf Optical Character Recognition (OCR) systems (e.g. OmniPage [**62**], Tesseract-OCR [**76**], ABBYReader [**1**]) are designed to work on scanned document images with relatively clean background and uniform text patterns, and they could not obtain good recognition performance on text regions of scene images. Text recognition is implemented by STC segmentation and STC prediction. STC segmentation partitions a detected text region into multiple image patches, each of which contains only one text character (see Fig. 5–1). *STC prediction* is a multi-class classification within pre-defined sample space, predicting recognized code from extracted features of a character patch. In previous text recognition algorithms, STC segmentation and recognition were processed in three ways [**12**]. First, character-like properties were defined to dissect text regions into candidate patches, where STC prediction was then applied to. Second, text regions were densely searched for text component with high confidence score of one character class, and the confidence score was obtained from STC prediction. Third, lexical analysis is applied to directly infer the whole words from confused STC prediction within text regions. Above methods show that STC prediction would always play a significant role in text word recognition. Thus, an improvement of STC prediction will result in better performance of word/phrase recognition in text regions.

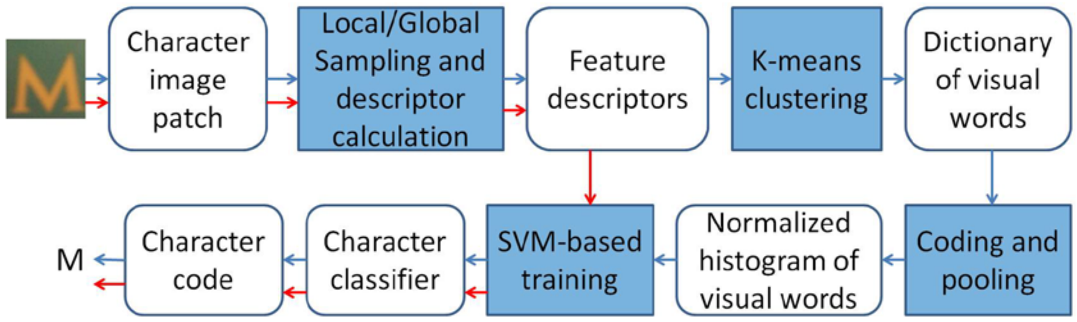


Fig. 5–1. Flowchart of STC prediction framework in performance evaluation, where the shadowed boxes denote the processing steps with multiple options and the white boxes denote the outputs of the processing steps. The blue arrows denote the processing of local sampling based feature representation, and the red arrows denote the processing of global sampling based feature representation.

To predict the category of an STC, we need a discriminative feature representation to model STC structure. Each STC patch is mapped into its feature representation in the form of a vector, which is regarded as a point in feature space. Then learning model is applied to train a robust character classifier, and perform multi-class classification to predict the category of a testing patch.

Most of the existing feature representations for STC prediction are generated from local sampling in Bag-of-Words (BOW) model. In this chapter, we summarize the categories of local-sampling and BOW based feature representation, and design better feature representation to improve STC prediction by using global sampling and Fisher vector. In local sampling, we detect key-points and compute local descriptors. In BOW model, we build dictionary of visual words, and perform feature coding and pooling to obtain a histogram of visual words, i.e., BOWs. To obtain feature representation from global sampling, we compute descriptor directly from the whole character patch without processing key points, dictionary, coding or pooling. Fig. 5–1 depicts the flowchart of feature representation from local and global sampling.

5.2. PREVIOUS WORK

A variety of feature representations for STC prediction were proposed. In [**93**], Gabor filter responses on synthetic STC were employed to extract features of character appearance. Then the results of STC prediction are combined with language, similarity and lexicon model to perform word-level recognition. In [**77**], SIFT descriptors were adopted to build a similarity expert to compute the character similarity, based on which integer program was applied for word recognition. In [**91**], HOG descriptors were densely extracted and cascaded as feature representations of character patches, and normalized cross correlation analysis of character similarity was used for STC prediction. In [**90**], Random Ferns algorithm was adopted for character detection, and pictorial structures with lexicon model were employed for word configuration and recognition. In [**57**], HOG feature was extracted for character recognition conditional random field was adopted to combine character detection and word-level lexicon analysis. In [**16**], local features of character patches were extracted by an unsupervised learning method related to a variant of K-means clustering, and spatially pooled by cascading sub-patch features. In [**59**], feature extraction for STC prediction was generated from Maximally Stable Extremal Regions (MSER), which is split into 8 levels by MSER boundary orientations. In [**105**], STC prediction for Chinese, Japanese and Korean characters was performed by Scale Invariant Feature Transform (SIFT) feature matching to template character patches, in which a voting and geometric verification algorithm was designed to remove false positive matches. However, most previous algorithms considered STC prediction as a small component of the whole framework of scene text information extraction. They focused more on lexicon-based word configuration and recognition, without complete quantitative analysis of image-based feature representation. However, most word-level processing depends on the results of character recognition, e.g., prediction score of character classifier. In this paper, we present performance evaluations on STC prediction under a general framework of object recognition, which consists of two processes: feature representation and multi-class classification.

5.3. Feature Representations for Predicting STC Categories

The most significant role in scene text recognition is to work out a multi-class classifier to predict the category of a given STC. This classifier relies on a feature representation that models the representative structure of each STC category and the discriminative structure between STC categories.

5.3.1. Low-level Feature Descriptor

Low-level features are extracted from STC image patches to describe appearance and structure of STCs from all 62 STC categories. Our framework involves 6 state-of-the-art feature descriptors which are applied to the key points sampled from STC image patches, including Histogram of Oriented Gradient (HOG) [**18**], Scale Invariant Feature Transform (SIFT) [**49**], Speed Up Robust Features (SURF) [**4**], DAISY [**83**], Binary Robust Independent Elementary Features (BRIEF) [**9**], and Oriented Fast and Rotated BRIEF (ORB) [**70**]. These feature descriptors have been commonly-used in the general visual recognitions, and many previous publications have demonstrated their effectiveness on object, texture, and scene recognitions.

In the implementations of these low-level, descriptors, HOG sets block size to be half of patch (or sub-patch) size, and block stride to be half of block size. Each block contains 4×4 cells, and the bin number of gradient orientations is 9. The other 5 descriptors are implemented by default parameters in public available source code and OpenCV2.4 [**64**]. We tried to tune the parameters, but did not obtain any apparent improvement in the two benchmark datasets CHARS74K and ICDAR2003.

In the design of feature representation, besides the choice of local feature descriptors, key-point sampling and feature coding/pooling play a significant role. We will present several methods of key-point sampling and schemes of feature coding/pooling.

5.3.2. Dense Sampling and Bag-of-words Modeling

Dense sampling is to extract key-points row-by-row and column-by-column under a stride from STC image patch. It covers more complete character structure, than sparse interest point detectors along with local feature descriptors that are widely used in image matching. In our experiments, dense sampling is designed as follows. Given a character patch, we first resize it into a square patch whose width equals to height, and then extend the side length into the nearest power of 2 (e.g., 128×128, 256×256). Next, in an $L \times L$ character patch, sub-patch in $(L/2) \times (L/2)$ is generated as feature window to extract feature descriptor, sliding from top-left to horizontal and vertical directions. The center of a sub-patch is regarded as a key-point, and the stride of two neighboring key-points is $L/8$ in both directions. Since L is a power of 2, we obtain $[(L/2)/(L/8) + 1] \times [(L/2)/(L/8) + 1] = 25$ key points from a character patch. The sub-patch at a key point is regarded as support region, generating a feature descriptor $x \in R^M$ where M denotes the dimensions. Since key point locations and sub-patch sizes are determined by the character patch size L, this local sampling method can be adaptive to scale changes of character patches.

Low-level feature descriptors are extracted from the sampled key-points of character patches, and we apply K-means clustering to build dictionary $D \in R^{M \times K}$, where M denotes the dimension of feature descriptors and K denotes the number of visual words. We use d_j to denote the j-th visual word. The dense sampling extracts 25 key-points from each character patch, so we generate a total of 23250 feature descriptors from CHARS74K and 154625 feature descriptors from ICDAR2003. To evaluate the impact of dictionary size on STC prediction, each type of feature descriptor at each dataset generates 5 dictionaries in different sizes. According to the number of feature descriptors from the datasets, we set the dictionary sizes to be 500, 1000, 2000, 3000, and 5000 respectively.

A number of low-level feature descriptors $\{x_i \mid 1 \leq i \leq N\}$ are extracted from a character patch, where N denotes the total number of descriptors. They are mapped into a histogram of visual words by coding and pooling [**48**]. The coding process is used to map each feature descriptor x_i into a histogram of visual words c_i based on the dictionary D. The state-of-the-art coding schemes include Hard Assignment (HARD), Soft Assignment (SOFT), and Sparse Coding [**98**] (SC) as Eq. (5 – 1) in top-down order. The parameter β and γ in SOFT and SC are used to control softness and sparseness respectively.

$$c_{ij} = \begin{cases} 1 & if\ j = \underset{j=1,\dots,n}{\operatorname{argmin}} \left\| x_i - d_j \right\|_2^2 \\ 0 & otherwise \end{cases}$$

$$c_{ij} = \frac{\exp(\beta \left\| x_i - d_j \right\|_2^2)}{\sum_{k=1}^{N} \exp(\beta \| x_i - d_k \|_2^2)} \qquad (5 - 1)$$

$$c_i = \underset{c}{\operatorname{argmin}} \| x_i - Dc \|_2^2 + \gamma \| c \|_1$$

The pooling is employed to aggregate coded features c_i into the final BOW feature representation$p$. The popular pooling schemes include Average Pooling (AVE) and Max Pooling (MAX) as Eq. (5 - 2) in top-down order.

$$p_j = (1/N)\sum_{i=1}^{N} c_{ij}$$
$$p_j = \max_i c_{ij} \tag{5 - 2}$$

In BOW model, the final feature representation is denoted as $\Psi(X) = \{p_1, p_2, p_3, \dots, p_N\}$

5.3.3. Dense Sampling and Fisher Vector

To generate more discriminative feature representation, we employ the Fisher Vector to represent each STC based on the keypoints obtained from dense sampling. Fisher Vector provides a feature aggregation scheme based on the Fisher kernel which takes the advantage of both generative and discriminative models. Fisher Vector describes each feature descriptor using the deviation with respect to the parameters of a generative model.

Fisher Vector employs the Gaussian mixture model (GMM) as the generative model $U_\lambda(x) = \sum_{k=1}^{K} w_k u_k(x)$, and u_k is the kth Gaussian component:

$$u_k(x) = \frac{1}{2\pi^{\frac{D}{2}}|\Sigma_k|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\,(x-d_k)'\,\Sigma_k^{-1}\,(x-d_k)\right\},$$
$$\forall k : w_k \geq 0,\ \sum_{k=1}^{K} w_k = 1. \tag{5 - 3}$$

where the feature descriptor $x \in \mathbb{R}^D$; K is the number of Gaussian components; w_k, d_k, and Σ_k correspond to the mixture weight, mean vector, and covariance matrix, respectively. We assume Σ_k to be a diagonal matrix with the variance vector σ_k^2. The parameters $\lambda = \{w_k, d_k, \Sigma_k,\ k = 1, \dots, K\}$ of GMM are estimated based on a large set of training low-level descriptors by the Expectation Maximization (EM) algorithm to optimize the Maximum Likelihood (ML).

For a set of descriptors $X = \{x_1, \dots, x_N\}$ extracted from a STC patch, the soft assignment of descriptor x_i to component k is defined as:

$$\gamma_i^k = \frac{w_k u_k(x_i)}{\sum_{j=1}^{K} w_j u_j(x_i)}. \tag{5 - 4}$$

The Fisher Vector representation of X is $\Psi(X) = \{\rho_1, \tau_1, \dots, \rho_K, \tau_K\}$, where ρ_k and τ_k are the D-dimensional gradients with respect to the mean vector d_k and the standard deviation σ_k of the kth Gaussian component:

$$\rho_k = \frac{1}{N\sqrt{w_k}} \sum_{i=1}^{N} \gamma_i^k \left(\frac{x_i - d_k}{\sigma_k}\right), \tag{5 - 5}$$

$$\tau_k = \frac{1}{N\sqrt{2w_k}} \sum_{i=1}^{N} \gamma_i^k \left[\frac{(x_i - d_k)^2}{\sigma_k^2} - 1\right]. \tag{5 - 6}$$

Compared to BOW-based representations, Fisher Vector has the following merits: (1) BOW is a particular case of Fisher Vector, i.e., the gradient to the component weights of GMM. The additional gradients with respect to the means and variances in Fisher Vector provide extra distribution information of descriptors in the low-level feature space. (2) The Fisher Vector can be computed upon a much smaller visual vocabulary which significantly reduces the computational cost. (3)

Fisher Vector performs quite well with simple linear classifiers which are efficient in both training and testing.

We follow the two normalization schemes introduced in [**68**], i.e., $L2$ and power normalization. The $L2$ normalization is used to remove the dependence on the proportion of class-specific information contained in a patch, in other words, to cancel the effect of different amount of foreground and background information contained in different images. The power normalization is proposed due to the fact that as the number of Gaussian components increases, Fisher Vector becomes peaky around zero in a certain dimension. This negatively impacts the computation of feature distance. The power normalization $f(z) = \mathrm{sign}(z)|z|^{\alpha}$ with $0 < \alpha \leq 1$ is applied to each dimension z in the Fisher Vector. We utilize $\alpha = 0.5$ (i.e., the Hellinger kernel) to compute the signed square-root. In our representation, we first apply the power normalization and then the L2 normalization.

In order to decorrelate the data to make it fitted more accurately by a GMM with diagonal covariance matrices, we apply a PCA on the SIFT descriptors to reduce them from $D = 128$ to 32. The number of Gaussian components K is empirically determined as 50. So each image patch of STC is represented as a feature vector with 3200 dimensions. SVM learning model is employed to generate a max-margin hyper plane in this feature space to classify the 62 STC categories. This hyper plane is defined as the STC predictor. The LIBLINEAR is used to implement SVM training and testing. Given an STC cropped from image frame, the STC predictor is able to compute its category from one of the 62 candidates as a label and output a 62-dimensional prediction scores as the probability of each category.

5.3.4. GLOBAL SAMPLING

Text characters are atomic objects, and a portion of a text character contains little information related to the whole structure. It is difficult to use part-based schemes like [**23**] to detect and recognize text. Thus global sampling is designed to use the whole character patch as a key-point neighborhood window to extract features. Key point detection, coding and pooling process are all skipped to largely reduce information loss. Compared with local sampling, the advantages of global sampling are two aspects. Firstly, there is no coding, so no information loss. Secondly, spatial structure is preserved when concatenating descriptors of grids in order.

Since most of local feature descriptors like SIFT, SURF and DASIY are usually paired with their respective key-point detectors, global sampling process adopts only HOG descriptor which does not require specific key-point detection. Thus we define it as GHOG. In Chapter 8, we will present the evaluation result of each feature representation and compare their performance under the same measurements.

5.4. Feature Representations for Predicting True Positive STCs from Background Outliers

As described in Chapter 4, feature maps and haar-like block patterns are able to model specific structure of text string fragment sample, for distinguishing text string from non-text background outlier. One positive fragment sample usually contains 2 ~ 4 different candidate character components, which have large intra-class variation. Some non-text background outliers may exist in text regions, and STC prediction cannot filter them out but only assign them to one of the 62 categories. To remove the possible background outliers before STC prediction, in this section we design a feature representation to distinguish single character from background outlier through a binary classification, which is named as *STC certification*.

First interest points and their neighboring windows are detected over a character sample. As shown in Fig. 5–2, multi-scale saliency detector [**38**] and Harris corner detector [**29**] are employed to detect interest points in a character sample. Multi-scale saliency detector prefers blob parts, such as strokes and holes, in a character. Harris corner detector prefers corner points around the ends of a stroke and the joints of neighboring strokes in a character. The salient and corner points correspond to characteristic components of STCs, so we can extract characteristic low-level features of STCs. However, these low-level features are still not discriminative enough to classify text characters from non-text background outliers, because the interest point windows cover only stroke fragments of the character sample, in the form of bars or arcs. These stroke fragments cannot provide spatial information of character structure. To solve this problem, we calculate the correlation between local interest points and their counterparts of the character sample, and generate more complex and characteristic structure (see Fig. 5–3), where discriminative features can be extracted to model character structure.

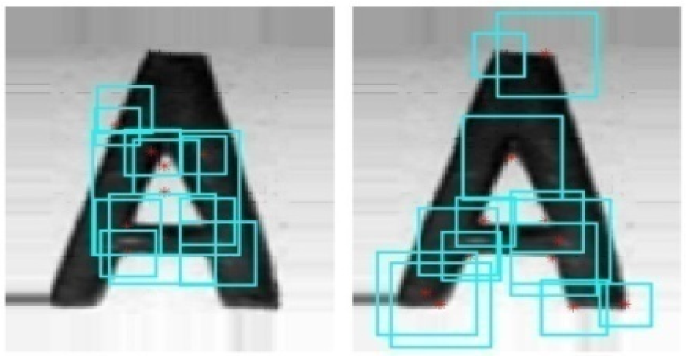


Fig. 5–2. Detected interest points in red and their neighboring windows in cyan. The left is obtained from multi-scale saliency detector and the right is obtained from Harris corner detector.

For an interest point window w_0, we use 3 symmetrical windows w_i $(1 \leq i \leq 3)$ as correlated counterparts of w_0 . They are generated with respect to horizontal midline, vertical midline and center of the image patch respectively, as shown in Fig. 5–3. To combine the structure layout of w_0 and its correlated counterparts, we define 4 types of structure correlations as Eq. (5–7) including sum, absolute difference, reciprocal of sum, and reciprocal of absolute difference. The 4 types of structure correlations could efficiently fuse the structure components of interest point window w_0 and its 3 symmetrical counterparts, and ensure the values of the fused windows within a reasonable range.

$$\boldsymbol{R} = \langle w_0 + w_i, \quad \frac{1}{w_0 + w_i}, \quad \|w_0 - w_i\|, \quad \frac{1}{\|w_0 - w_i\|} \rangle \tag{5–7}$$

Structure components generated by the fused windows can provide more discriminative low-level features of STC, although these components do not belong to original character structure. Then state-of-the-art low-level feature descriptors, HOG [**18**], SIFT [**49**] and LBP, are employed to extract appearance features from the fused windows.

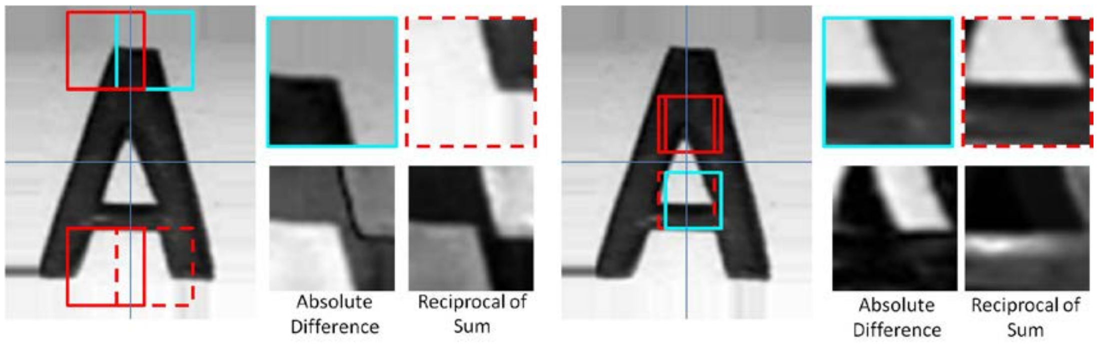


Fig. 5–3. Two structure correlation examples. The cyan squares represent original detected interest point windows, and the red squares represent their corresponding symmetric counterparts. We extract the original windows and one of their symmetric counterparts (marked by red dash line), and calculate their correlated windows under absolute difference and reciprocal of sum.

The low-level features are then input into BOW model for coding and pooling processes, obtaining feature representation in the form of visual word histogram. The visual word histograms are further processed according to specific structure of STC, and then input into SVM model to train a binary classifier that distinguishes single character from background outlier. A comparative experiment (see Section 8.2.1) proves that structure correlation can improve the performance of text classification.

STC certification can improve the performance of scene text extraction only if text information in camera-based scene image appears in enough resolution, because it is difficult to distinguish a low-resolution character from background outlier. Therefore, our prototype system in real applications skips this step to improve the efficiency of the whole system.

5.5. Learning Process

STC prediction depends on SVM-based training and testing over the STC samples. While the learning process in scene text detection is to select the representative combinations of feature maps, haar-like block patterns and calculation schemes to distinguish text from non-text, the learning process in STC prediction treats the feature representation vector of an image patch as a point in feature space, which describes the STC structure in that patch. Thus we would adopt SVM learning model to generate hyper-planes in feature space as STC classifier, rather than the Adaboost algorithm to select optimized combinations of the weak classifiers.

In the SVM-based learning process, we adopt multiple SVM kernels, including Linear Kernel and χ^2 Kernel, to evaluate the feature representations of STC structure. A brief introduction of SVM model will be presented in detail in Appendix A.2.

5.6. Summary of Contributions

In this chapter, our main contributions are to find out discriminative feature representations to model scene text character structure for recognition task, on the basis of state-of-the-art low-level descriptor and Bag-of-Words model and Fisher vectors. In each step of designing the feature representation, we adopt multiple available algorithms. Then a comparative study is performed to find out the best combinations of low-level descriptors, BOW dictionary sizes, and coding/pooling schemes.

Chapter 6 Word-level Configuration of Scene Text by Conditional Random Field

6.1. Problem Formulation

Due to the cluttered background noise and multiple STC patterns in natural scene, the accuracy of STC prediction is limited. It only takes account of the appearance of a single STC in an image patch, but ignores its context information of neighboring STCs. Besides, it does not involve any lexical analysis and word recognition only based on one-by-one single STC prediction ignores the possible constraints of STC combinations in the lexicon model. Moreover, some STCs tend to be incorrectly classified to other categories in a similar structure, e.g., letter "O" and digit "0". Therefore the STC prediction can only be considered as preliminary results of word recognition in the character level. As shown in Fig. 6–1, if STC prediction can obtain perfect results, word-level recognition is also perfect by just listing all the results of STC predictions. However, when STC prediction is not perfect, the performance of word recognition will seriously decrease.

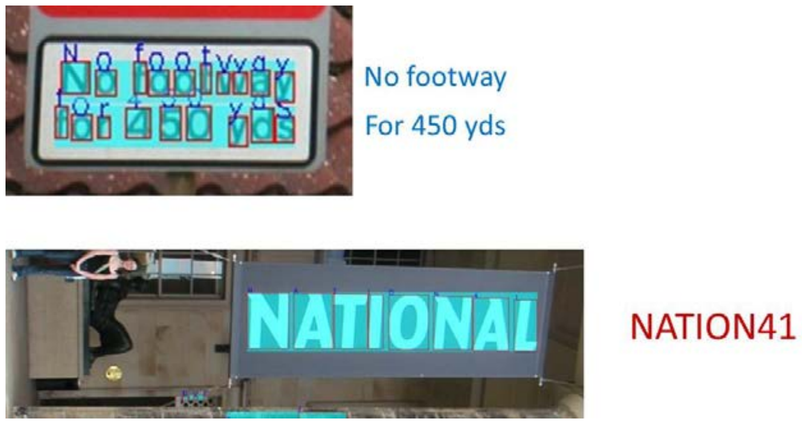


Fig. 6–1. Top row demonstrates the word recognition based on perfect STC predictions, and bottom row demonstrates the error of word recognition because of slight error of STC predictions.

6.2. CONDITIONAL RANDOM FIELD

To rectify STC prediction and obtain recognized words compatible with dictionary, CRF model [**92**] [**90**] [**57**] [**75**] was usually adopted to configure text word from predicted STCs and the resulting prediction score in SVM. CRF model is a discriminative undirected probabilistic graphical model $\langle V, E\rangle$ and encodes both the relationships between observations and category labels of nodes as well as the relationships between neighboring nodes in the graphical model. In the notations $\langle V, E\rangle$, V denotes the node set and E denotes edge set.

In the potential results of word recognition, we define each STC prediction as a random variable V_i as a node in CRF model. STC prediction assigns a category label to each node, which can also be considered as assigning an observation to each random variable. We set the total number of STC predictions, which is also the total number of nodes as $|V|$. The cost function of a CRF model is defined as Eq. (6–1).

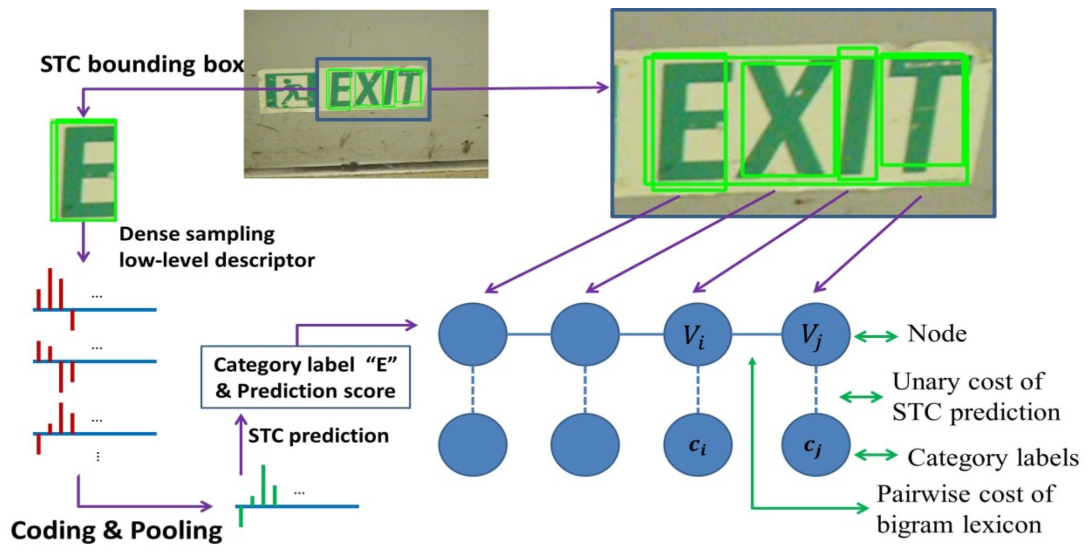
$$L(\boldsymbol{V} = \boldsymbol{c}) = \sum_{i \in V} \mu_i S(V_i = c_i) + \sum_{(i,j) \in E} \lambda_{ij} T\big(V_i = c_i, V_j = c_j\big) \tag{6–1}$$

where $\boldsymbol{V}$ is the set of all nodes, $\boldsymbol{c}$ represents their corresponding category label, S is the cost function of single node to measure the suitability of category labels obtained from STC prediction, T is the cost function of an edge to measure the compatibility of neighboring category labels obtained from STC predictions. In word recognition, CRF model always appears in the form of chain, that is, each pair of neighboring characters are connected by an edge. μ_i and λ_{ij} are parameters obtained from training data.

CRF model provides a framework to build probabilistic model and label sequence nodes. The concrete cost functions of node and edge in CRF can be customized according to different problems. In our framework, the involved cost functions are defined as:

$$S(V_i = c_i) = 1 - \mathbf{norm}(Score(V_i))$$
$$T(V_i = c_i, V_j = c_j) = 1 - Freq(c_i, c_j) \quad (6–2)$$

where $S(V_i = c_i)$ is unary cost of a node, $T(V_i = c_i, V_j = c_j)$ is pairwise cost of neighboring nodes, $Score(V_i)$ represents the STC prediction scores at node X_i, $\mathbf{norm}(\cdot)$ represents normalization and $Freq(c_i, c_j)$ represents the frequency of the bigram lexicon c_i and c_j. In our experiments, the bigram lexicon frequency is generated from all ground truth words of ICDAR-2003, ICDAR-2011 and Street View Text Datasets. In future work, we will design more robust cost functions to further improve word recognition under CRF model.



STC bounding box
EXIT
E
Dense sampling low-level descriptor
Category label "E" & Prediction score
STC prediction
Coding & Pooling
V_i
V_j
c_i
c_j
Node
Unary cost of STC prediction
Category labels
Pairwise cost of bigram lexicon

Fig. 6–2. On the left, STC is extracted by detection and tracking, and then transformed into a feature representation through low-level feature descriptor and coding & pooling processes. SVM-based STC predictor is applied to obtain its category label and prediction score. On the right, in a tracking frame of scene text, each STC bounding box is defined as a node in CRF model. Unary cost of STC prediction and pairwise cost of bigram lexicon is defined in this graphical model.

Fig. 6–2 illustrates the CRF model $\langle \boldsymbol{V}, \boldsymbol{E} \rangle$ of an image patch cropped from a video frame. Each STC is defined as a random variable node in CRF, and each pair of neighboring STC nodes is mutually connected by an edge. CRF model will generate proper μ_i and λ_{ij} parameters from training data, minimizing the cost function Eq. (6–1). Given a word in the form of scene text, if each character of this word has been assigned a score by STC prediction, the learned CRF model will assign compatible category labels $\boldsymbol{c}$ to nodes $\boldsymbol{V}$. In this process, some incorrect STC predictions can be rectified if they lead to high cost. For example, as shown in Fig. 6–1, the word "NATIONAL" is recognized as "NATION41" because the structures of "A" and "4" and the structures of "L" and "1" are very similar. STC predictions cannot distinguish them by vision-based feature representation. But in CRF model, the combination "N4" and "41" results in larger edge cost because the frequencies of the two bigrams are lower than "NA" and "AL" respectively.

6.3. Summary of Contributions

In this chapter, our main contribution is to adopt the CRF model to perform word-level recognition. CRF model combines vision-based STC recognition and bigram frequency based lexical analysis. It is able to generate optimized labels that minimize the node costs and edge costs in CRF model. CRF is able to correct the STC recognition results.

Chapter 7 SCENE TEXT DETECTION

7.1. DATASETS

We evaluate the performance of our proposed framework on three benchmark datasets of scene images, ICDAR 2003 Robust Reading Dataset [**31**], ICDAR 2011 Robust Reading Dataset [**32**], Born-Digital Images Dataset [**33**], and Street View Dataset [**89**].

ICDAR-2003 and ICDAR-2011 are collected for robust reading competitions, and annotated text regions. ICDAR 2003 robust reading dataset contains about 500 scene images and 2258 ground truth text regions in total. In our experiments, the scene images containing non-text or only a single character are excluded. Thus 487 scene images are used for performance evaluation. The involved image sizes range from 640×480 to 1600×1200.

ICDAR-2011 robust reading dataset contains 484 scene images with 848 ground truth text regions in total, in which 229 images are for training and 255 images for testing in ICDAR 2011 Robust Reading competition. We evaluate the framework on all the images containing text strings with no less than two character members. The image size ranges from 422×102 to 3888×2592. The proposed framework is applied on the above datasets for text localization. The localization processes are carried out in each scene image and its inverse image, and the results are combined to calculate the localized text regions.

Born-digital images and broadcast video images are also used to evaluate our framework. Born-digital images are electrical documents with colorful captions and illustrations. Mostly they exist in web pages, book covers, and posters. In born-digital images, text characters and strings are more colorful. Besides, born-digital image has higher frequency of occurrences of text and smaller character sizes than scene image. A dataset of born-digital images is released for ICDAR-2011 robust reading competition [**33**]. It contains 420 born-digital images with ground truth text regions. The average image size is about 352×200.

The Street View Text Dataset [**89**] is collected from Google street view. This dataset is more challenging because it is captured from outdoor environments with illumination variations. The text characters usually have low resolutions and are embedded into complex background outliers. It contains about 350 scene images and 900 ground truth text regions in total. In Street View Dataset, due to more complex background interferences, more false positive detections are generated, so the precision is much lower than that in ICDAR Robust Reading Dataset.

7.2. EVALUATION MEASURES

Evaluation results are obtained from the comparisons between a group of detected text regions and ground truth text regions from manual labeling. The overlaps between detected regions and ground truth regions are defined as hit regions, which mean the correct detections. Then we define the area of a text region as the number of pixels in the region.

Based on these measures, *Precision* is defined as the ratio between the area of hit regions and the area of the detected regions. It is used to sense the amount of false positives in the detected regions. *Recall* is defined as the ratio between the area of hit regions and the area of the ground truth regions. It is used to sense the amount of missing detections in the ground truth regions. Then they are combined by harmonic mean to obtain *f-measure* as Eq. (7–1).

$$f\ mesure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7–1}$$

7.3. Evaluating Scene Text Detection by Text Layout Analysis on ICDAR Datasets

At first, we evaluate the performance of text layout analysis without considering the structural feature analysis. Experimental results on the Robust Reading dataset demonstrate that the combination of color-based partition and adjacent character grouping (CA) achieves the highest precision and recall. In most of the cases, color uniformity acts as a stronger indicator to distinguish the connected components of text characters from surrounding background. However color-based partition takes more computing time than gradient-based partition. Also color-based partition makes adjacent character grouping be performed in each of the color layers. Color-based partition still performs better when adjacent character grouping is replaced by the text line grouping. Text line grouping gives lower efficiency and precision than the adjacent character grouping for either partition. Adjacent character grouping is supported by the information of text orientations while text line grouping is performed for arbitrary text orientations, so its calculation cost is more expensive. Meanwhile, the indetermination of text orientation produces more false positive fitted lines.

By comparison with the algorithms presented in the text locating competition in ICDAR-2003, the precision of our algorithm achieves the first rank while the recall and *f*-measure is comparable with the algorithms with the high performance, as shown in Table 7–1.

Table 7–1. The comparison between our algorithm and the text detection algorithms presented in [**51**] [**50**] on the Robust Reading Dataset.

	Precision	Recall	*f*-measure
Ours	**0.71**	**0.62**	**0.62**
H. Becker	0.62	0.67	0.62
A. Chen	0.60	0.60	0.58
Ashida	0.55	0.46	0.50
HWDavid	0.44	0.46	0.45
Q. Zhu	0.33	0.40	0.33
Wolf	0.30	0.44	0.35
J. Kim	0.22	0.28	0.22
Todoran	0.19	0.18	0.18
N. Ezaki	0.18	0.36	0.22

7.4. Evaluating Scene Text Detection by Structural Feature Maps on Image Patches from ICDAR Datasets

Five feature maps of string fragment sample, including gradient magnitude, gradient orientation, stroke width, stroke orientation, and stroke width consistency, are measured in our experiments. We estimate the performance of each feature map by generating an SVM-based classifier of string fragments and evaluating its classification accuracy.

To train robust text classifier, we collect a training set of string fragments, which consists of 2000 representative positive samples and 2000 representative negative samples. We perform two experiments to evaluate our designed feature maps. First, the feature is evaluated within the collected training set of string fragments. The 2000 positive samples and 2000 negative samples are equally divided into two subsets respectively, one of which is used for classifier training and the other is used for evaluation. Second, the classifier is learned from the collected training set, and then evaluated on about 18,000 image patches, which are obtained from layout analysis on the scene images of ICDAR 2003 robust reading dataset.

Fig. 7–1 illustrates the evaluation results of the two experiments respectively, where hit rate represents the ratio of correctly classified samples in positive set, and false positive rate represents the ratio of incorrectly classified samples in negative set. The two figures demonstrate that stroke width consistency is more robust than the other features of text. Gradient magnitude and stroke width achieve comparable performance with stroke distribution and gradient orientation in the first experiment, but they become inferior in the second experiment. It is inferred that gradient orientations and stroke distributions are normalized into the ranges $(-\pi, \pi]$ and $[0, 1]$ respectively, so they are more robust to the variations of large number of samples in the second experiments. In our framework, all the feature maps are combined to model string fragments, because both figures show that the best performance of string fragment classification is achieved when combining all the features.

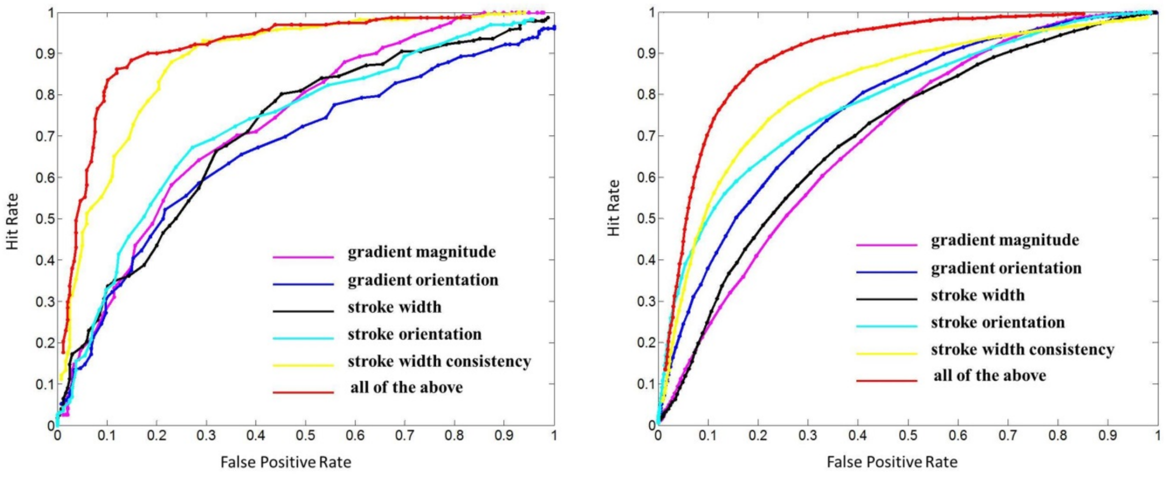


Fig. 7–1. Evaluation results of the 5 Gabor-based features on ICDAR-2003 robust reading dataset of scene images. Here hit rate is the ratio of correctly classified samples in positive set, and false positive rate is the ratio of incorrectly classified samples in negative set.

7.5. EVALUATING SCENE TEXT DETECTION BY COMBING LAYOUT ANALYSIS AND STRUCTURAL ANALYSIS

We combine layout analysis and structural analysis to improve the performance of scene text detection. ICDAR-2003 and ICDAR-2011 Datasets are respectively used for performance evaluation.

Table 7–2 presents the performance comparisons between our framework and the localization algorithms involved in ICDAR-2003 dataset [**31**]. It shows that the proposed framework outperforms most previous localization algorithms. Fig. 7–2 illustrates some example of text localization in ICDAR-2003 where the text regions are marked in cyan boxes.

Table 7–2. The comparison between our framework and the text localization algorithms presented in [**22**] [**51**] [**50**] on the Robust Reading Dataset.

Method	***Precision***	***Recall***	***f-measure***
Ours	**0.73**	**0.67**	**0.66**
B. Epshtein	0.73	0.60	0.66
H. Becker	0.62	0.67	0.62
C. Yi	0.71	0.62	0.62
A. Chen	0.60	0.60	0.58
Ashida	0.55	0.46	0.50
HWDavid	0.44	0.46	0.45
Wolf	0.30	0.44	0.35
Q. Zhu	0.33	0.40	0.33

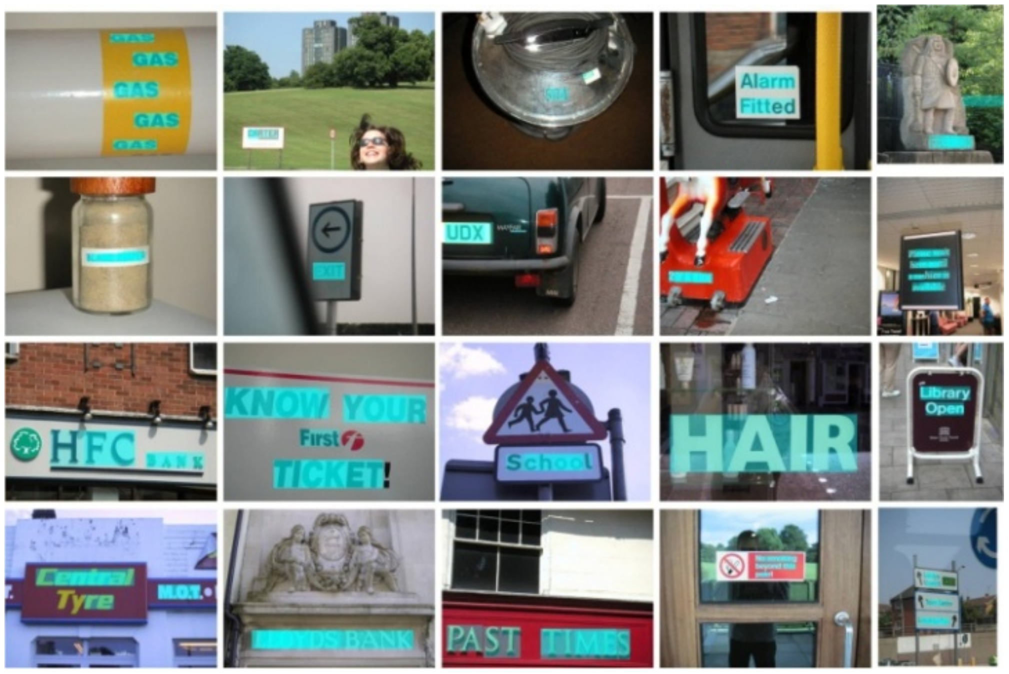


Fig. 7–2. Example results of text localization in the ICDAR-2003 Dataset, where the text regions are marked by cyan boxes.

Table 7–3 presents the performance comparisons between our framework and the localization algorithms involved in ICDAR-2003 dataset [**31**]. It shows that the proposed framework outperforms most previous localization algorithms. Fig. 7–3 illustrates some example of text localization in ICDAR-2003 where the text regions are marked in cyan boxes.

Table 7–3. The results of ICDAR-2011 Robust Reading Competition on Scene Text Localization (%) [**72**]. Our proposed framework won 2nd place.

Method	***precision***	***recall***	***f-measure***
Yi (2012)	**81.00**	**72.00**	**71.00**
Kim	62.47	82.9871.	71.28
Yi (2011)	**58.09**	**67.22**	**62.32**
TH-TextLoc	57.68	66.97	61.98
Neumann	52.54	68.93	59.63
TDM_IACS	53.52	63.52	58.09
LIP6-Retin	50.07	62.97	55.78
KAIST AIPR	44.57	59.67	51.03

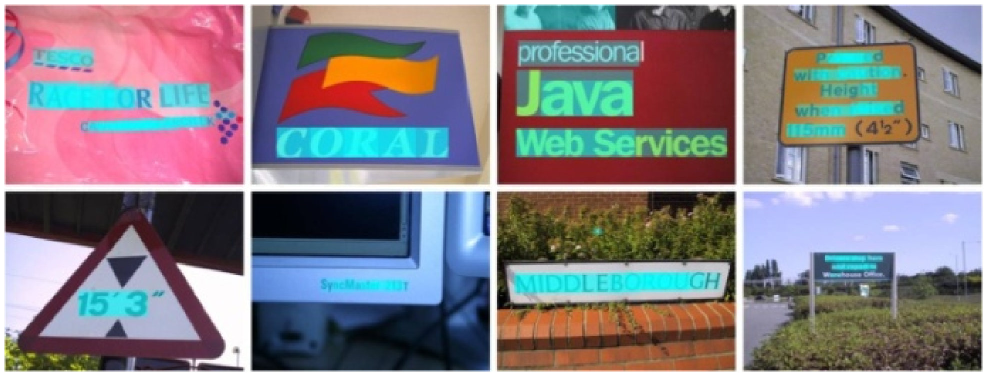


Fig. 7–3. Some example results of text localization in the ICDAR-2011 Dataset, where the text regions are marked by cyan boxes.

7.6. EVALUATING SCENE TEXT EXTRACTION ON BORN-DIGITAL IMAGES AND VIDEO IMAGES

We further evaluate our framework to extract text information from born-digital images and broadcast video images. Born-digital images are electrical documents with colorful captions and illustrations. Mostly they exist in web pages, book covers, and posters. In born-digital images, text characters and strings are more colorful. If we adopt color-based partition to process the scene image, the initial number of Gaussian mixtures in bigram color reduction is set as $K = 7$. Besides, born-digital image has higher frequency of occurrences of text and smaller character sizes than scene image. Thus in layout analysis we consider some connected components directly as string fragments and slice the corresponding image patches vertically to overlapped partitions with width-to-height ratio 2:1.

A dataset of born-digital images is released for ICDAR 2011 robust reading competition [**33**]. It contains 420 born-digital images with ground truth text regions. The average image size is about 352×200. We evaluate our framework by using the same measures on this dataset. Fig. 7–4 presents some examples of localized text regions in born-digital images.

Moreover, our framework is evaluated on broadcast video images. In most video images, text serves as titles and captions to introduce the content of television program. It is distributed on the top or bottom of the screen. The characters and strings also have the features of bigram color uniformity, stroke width consistency, and character alignment. Different from scene images, most text information in broadcast video image is subsequently added for audience reading, so they generally encounter fewer background interferences and pattern variations. Fig. 7–5 depicts some results of localization in broadcast video images.

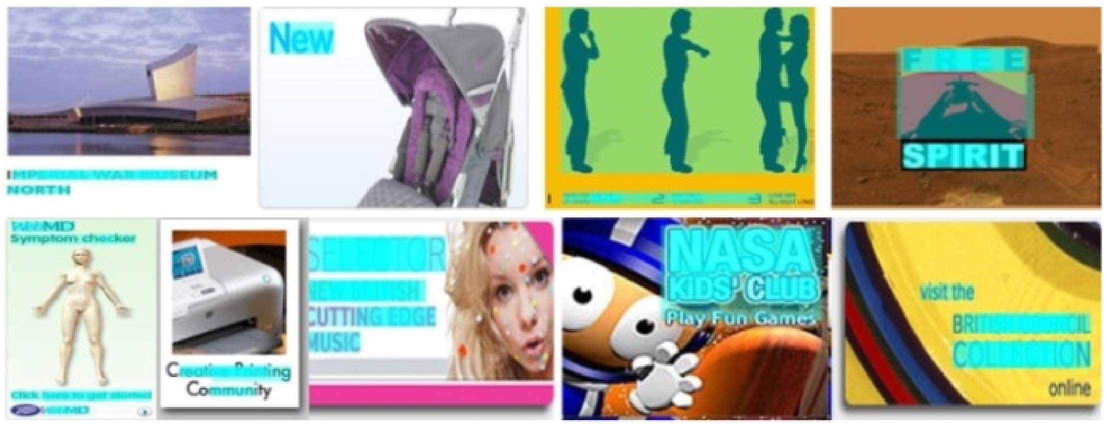


Fig. 7–4. Example results of text localization in the born-digital images.

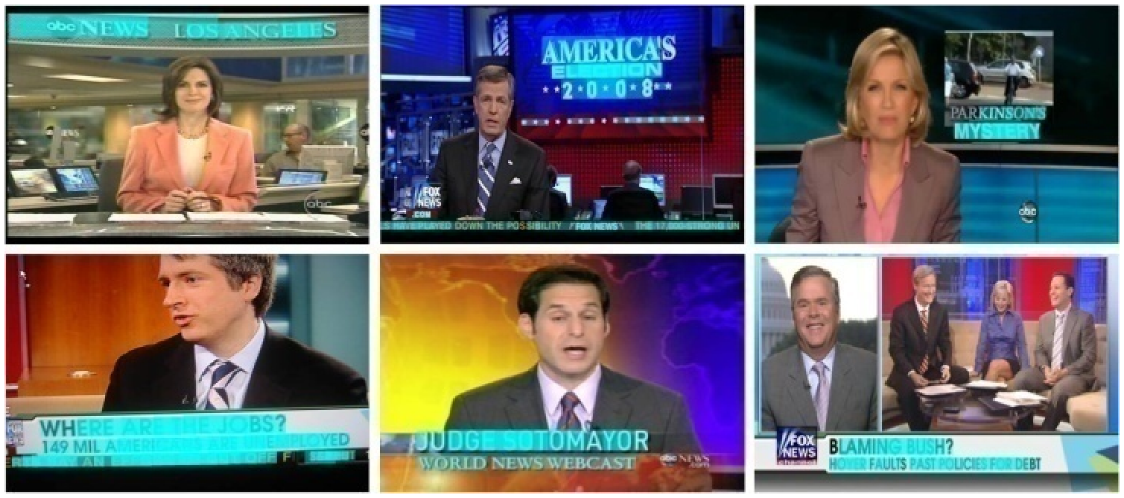


Fig. 7–5. Example results of text localization in broadcast video images.

7.7. Some Challenging situations to be addressed

Although our framework is able to detect most text information from complex background in natural scene, there are still some challenging situations that our framework cannot well handle. The main reason is that these situations do not satisfy our assumption of text layout and structure. Fig. 7–6 depicts some examples that our method cannot handle to locate the text information because of very small size, overexposure, characters with non-uniform colors or fade, strings with less than 3 character members, and occlusions caused by other objects such as wire mesh.

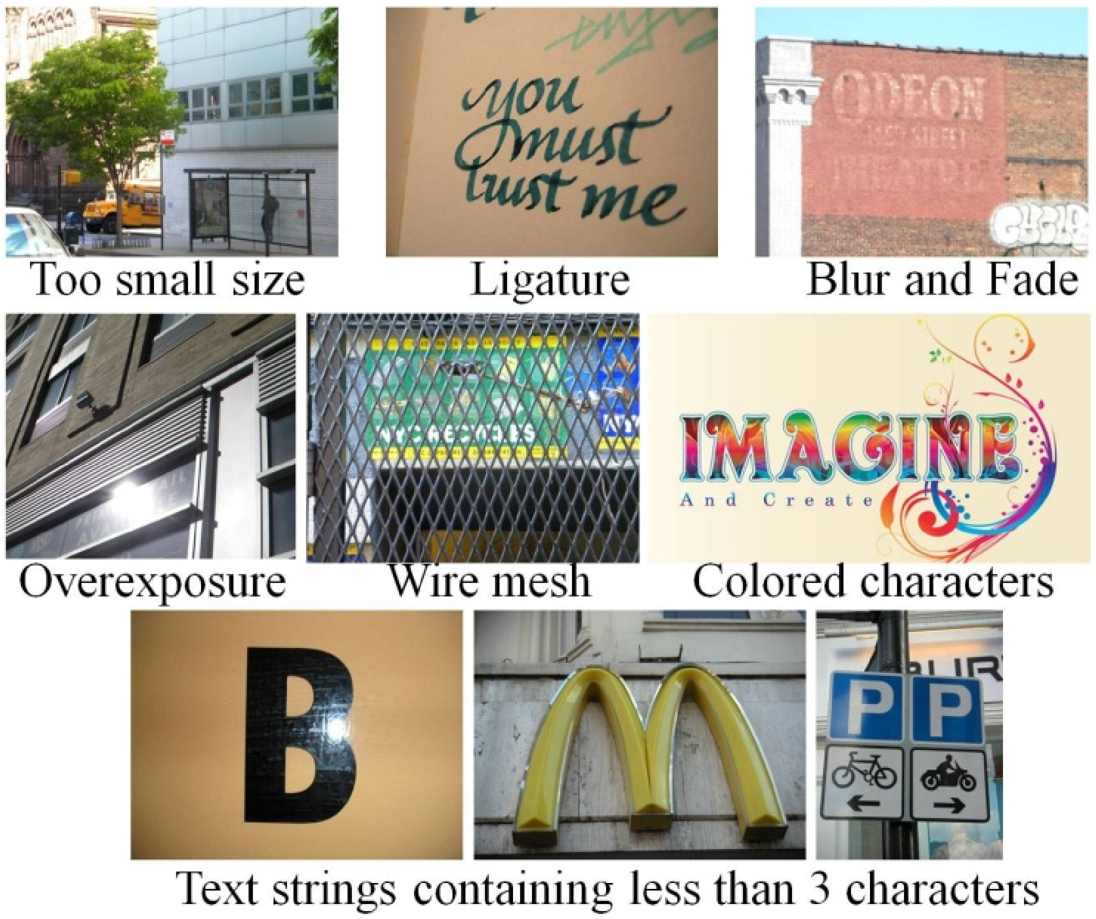


Fig. 7–6. Examples of images where our method fails.

7.8. SUMMARY OF CONTRIBUTIONS

In this chapter, the proposed methods of scene text detection are evaluated over benchmark datasets. The experimental results demonstrate the effectiveness of our method in handling complex background and multiple text patterns. From the perspective of quantitative analysis, our method outperforms the state-of-the-art results in the benchmark datasets.

Chapter 8 SCENE TEXT RECOGNITION

8.1. DATASETS AND EVALUATION MEASURES OF SCENE TEXT CHARACTER PREDICTION

The benchmark datasets prepared for scene text detection and recognition are mostly composed of natural scene images with text information. The image regions containing text are provided as ground truth labels. However, most of them are word-level text regions, without dissecting them into image patches of single characters. To evaluate STC prediction, two public datasets, CHARS74K [**19**] and ICDAR2003CH [**51**], are adopted in our experiments.

In CHARS74K dataset, there are three types of text characters, image-based characters cropped from natural scene image, hand-written characters, and computer-generated font characters. The first type is used for evaluating our STC prediction task. The characters from this type contains 62 character categories, i.e., digits *0~9*, English letters in upper case *A~Z*, and lower case *a~z*. The 62 character categories have nearly balanced numbers of character patches.

ICDAR2003CH dataset was used for the Robust Reading Competition of scene text detection and recognition. It contains 509 scene images and 2268 word-level text regions. These text regions have been partitioned into 11615 image patches of characters, obtaining 6185 training patches and 5430 testing patches. They cover all the 62 character categories, but the numbers of character patches between different classes are imbalanced.

Fig. 8–1 illustrates some examples of STCs cropped from text regions. We can observe the STCs have irregular patterns and similar structure to each other. The performance of STC prediction is measured by the average accuracy rate, i.e., the ratio of correctly predicted STCs in the testing set.

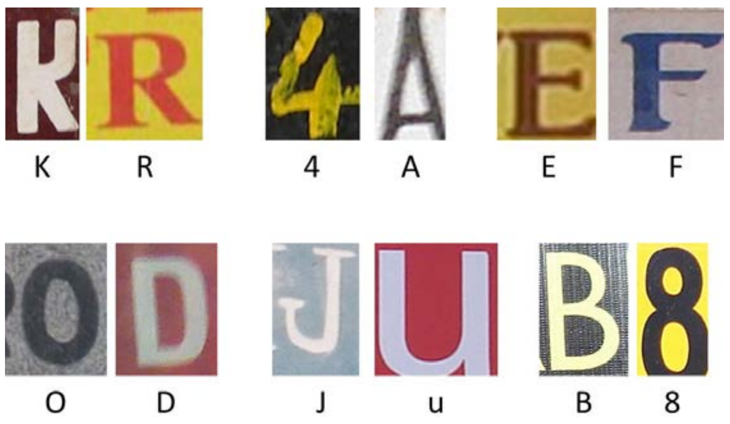


Fig. 8–1. Some examples of STCs cropped from text regions. Most STCs have similar structure to another counterparts.

8.2. EVALUATING FEATURE REPRESENTATIONS OF DENSELY LOCAL SAMPLING AND BOW MODELS

8.2.1. EVALUATING LOW-LEVEL DESCRIPTORS

Fig. 8–2 demonstrates performance evaluations of the 6 types of feature descriptors under dense sampling. Under HARD-AVE scheme, SURF obtains the best performance and HOG obtains the close second best. Under SOFT-AVE and SOFT-MAX schemes, HOG obtains the best performance.

BRIEF and ORB could obtain good performance in recognizing texture-rich object. The simple binary tests between pixels in a local support region in BRIEF and ORB is not well adapted to character recognitions because binary tests from uniform intensity regions (frequent in character patches) are not able to provide sufficient discriminative information. Fig. 8–2 also shows that SOFT-MAX scheme obtains better performance than SOFT-AVE and HARD-AVE schemes.

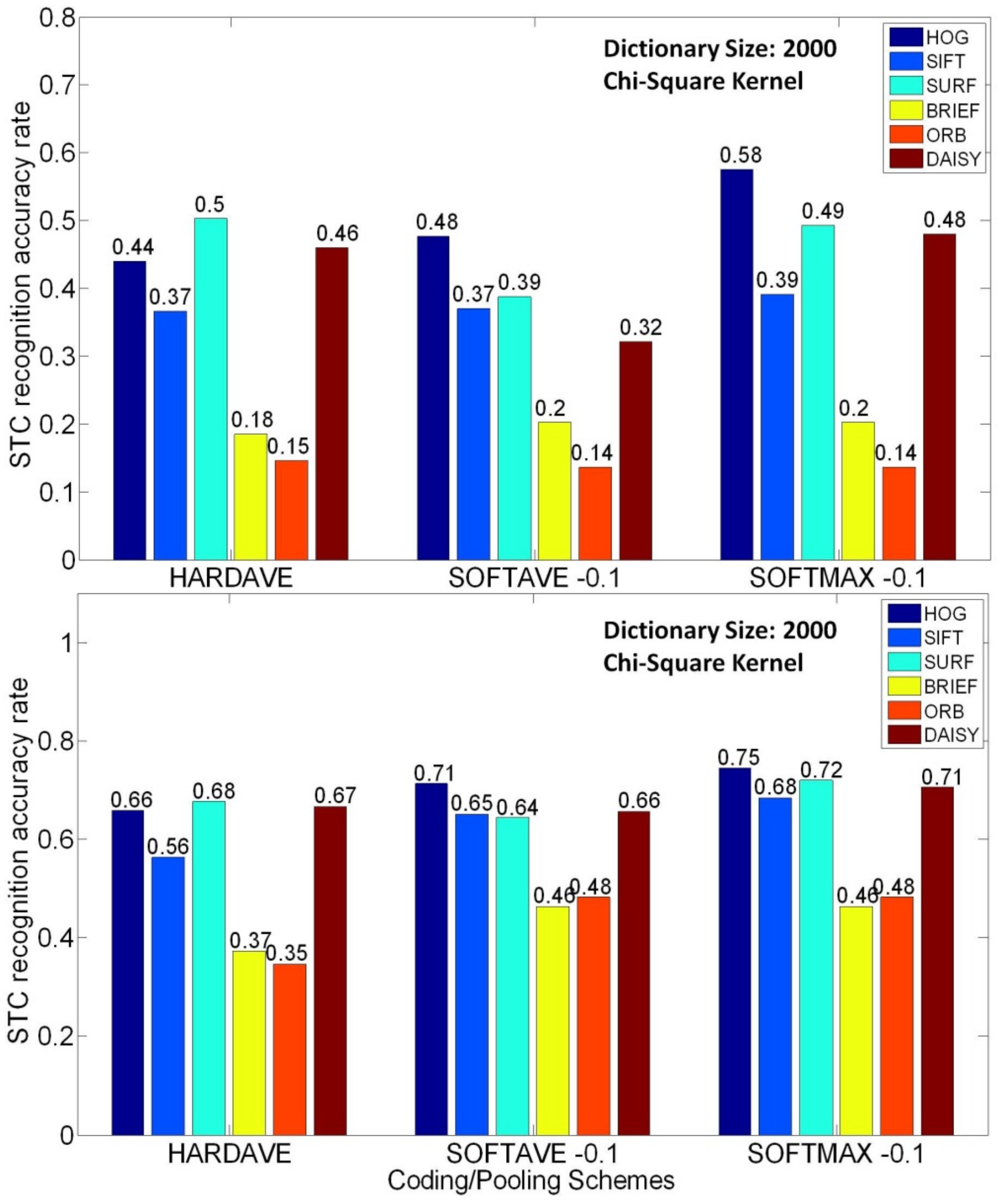


Fig. 8–2. Performance evaluations of 6 types of feature descriptors under three coding/pooling schemes, dictionary size 2000 and Chi-square SVM kernel. The value -0.1 in horizontal axis denotes β value in Eq. (5 – 1). The top figure shows results from CHARS74K and the bottom figure shows results from ICDAR2003CH.

As described in Section 5.4, a feature representation based on structure correlation of scene text character is designed to distinguish text character from background outlier. Over a set of image patches, in which text characters from ICDAR2003CH are used as positives and the other non-text patches are used negatives, an experiment is carried out to compare the performance of text character classification. We extract two types of low-level features from each sample, with and without structure correlations respectively. Three types of low-level features are adopted in this experiment. Then the low-level features are projected into respective visual word histograms. SVM-based cross-validation is performed to train classifier for distinguishing text character from background and evaluate the classifier. By setting thresholds of the SVM prediction scores, we generate curves in Fig. 8–3, in which the performance is measured by recall and precision. We define recall as the ratio between the number of correctly predicted positives and the total number of truth positives, and define precision as the ratio between the number of correctly predicted positives and the total number of samples being predicted as positives. The results show that structure correlation is able to generate more discriminative feature representations.

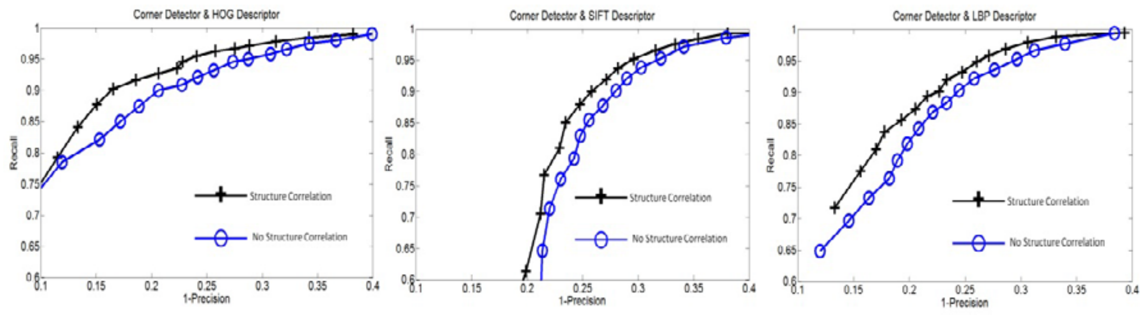


Fig. 8–3. Evaluation results of text classification. Text descriptors derived from structure correlation perform better than those directly derived from interest point windows. In addition, text descriptor presented by left figure is calculated from appearance features under Harris-Corner detector and HOG descriptor, and the right one is obtained from appearance features under Harris-corner detector and SIFT descriptor.

8.2.2. Evaluating Dictionary Size in BOW Model

The evaluation results in Fig. 8–4 show the relationship between STC prediction rate and the dictionary size. We can get some insights of the relationship between dictionary size and STC prediction accuracy.

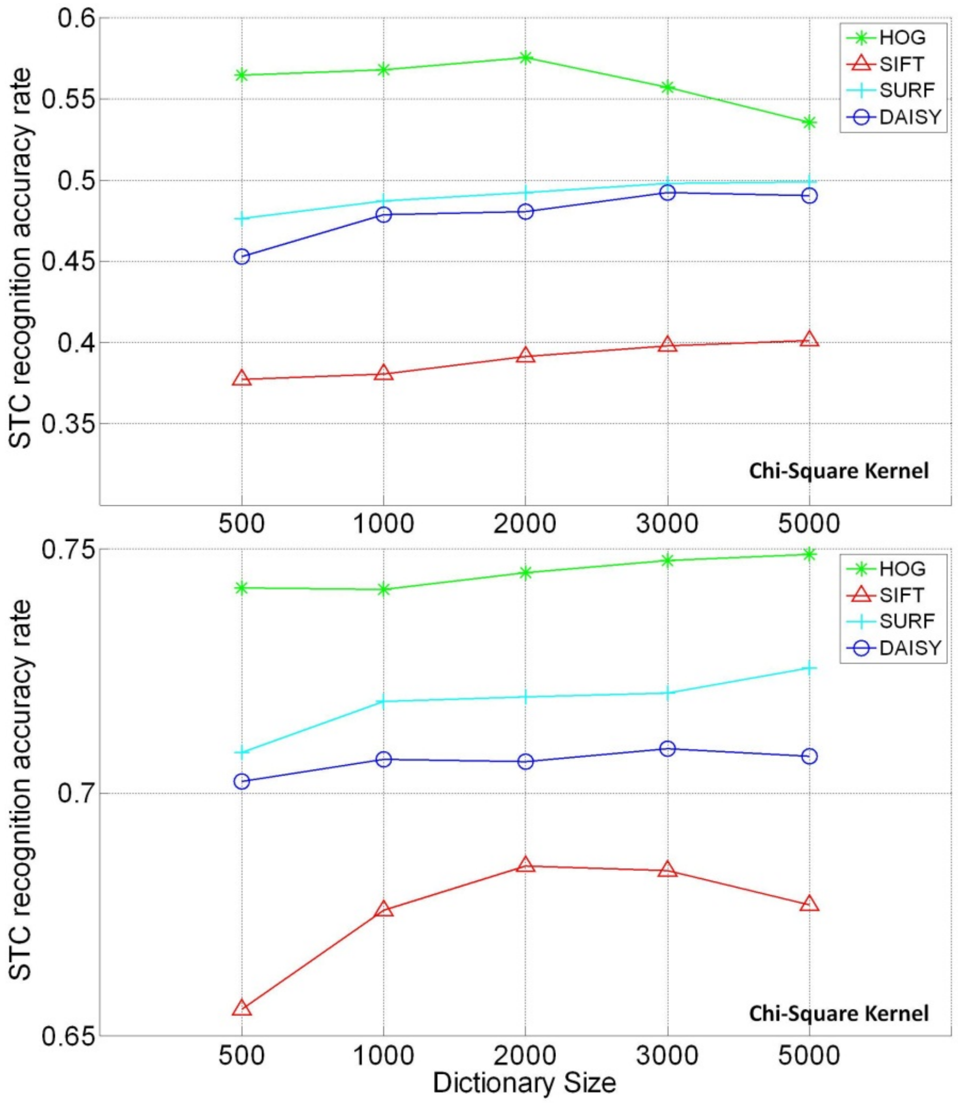


Fig. 8–4. Performance evaluations of 5 dictionary sizes under four feature descriptors and Chi-Square SVM kernel. Top figure results from CHARS74K and bottom figure results from ICDAR2003CH.

When the dictionary size is less than 2000, the performance is increased along the dictionary size change. However, the growth trend will cease when the dictionary size reaches a certain level. Then the performance will keep approximately consistently or slightly decrease. Compared with general object recognition in multiple scales and view angles in scene image, STC in image patches have relatively stable structure since the scale has been normalized with the STC patch size and the view angle does not largely change STC appearance. Thus this amount of visual words is sufficient to represent local features extracted from STC, and growth saturation of STC prediction performance on dictionary size is reached more rapidly.

8.2.3. EVALUATING CODING POOLING SCHEME

Fig. 8–5 depicts the performance evaluations of different coding and pooling schemes. In SOFT, we obtain two groups of results by setting the parameter β in Eq. (5 – 1) as -0.1 and -1 respectively. In SC, we set the parameter γ in Eq. (5 – 1) as 0.15. Currently, SC is not applied to ICDAR2003CH because of its high computational cost in coding optimization. In coding schemes, SOFT and SC are comparable, and both obtain better performance than HARD. As shown in Eq. (5 – 1), this is probably because the extreme sparseness of codes generated by HARD (only one coefficient per code is non-zero) might be ill-suited to character images, and SOFT and SC loose the constraint to alleviate information lose. In pooling schemes, MAX always obtains better performance than AVE, because maximum value usually contains the most significant information and its statistical properties make it well adapted to sparse representations.

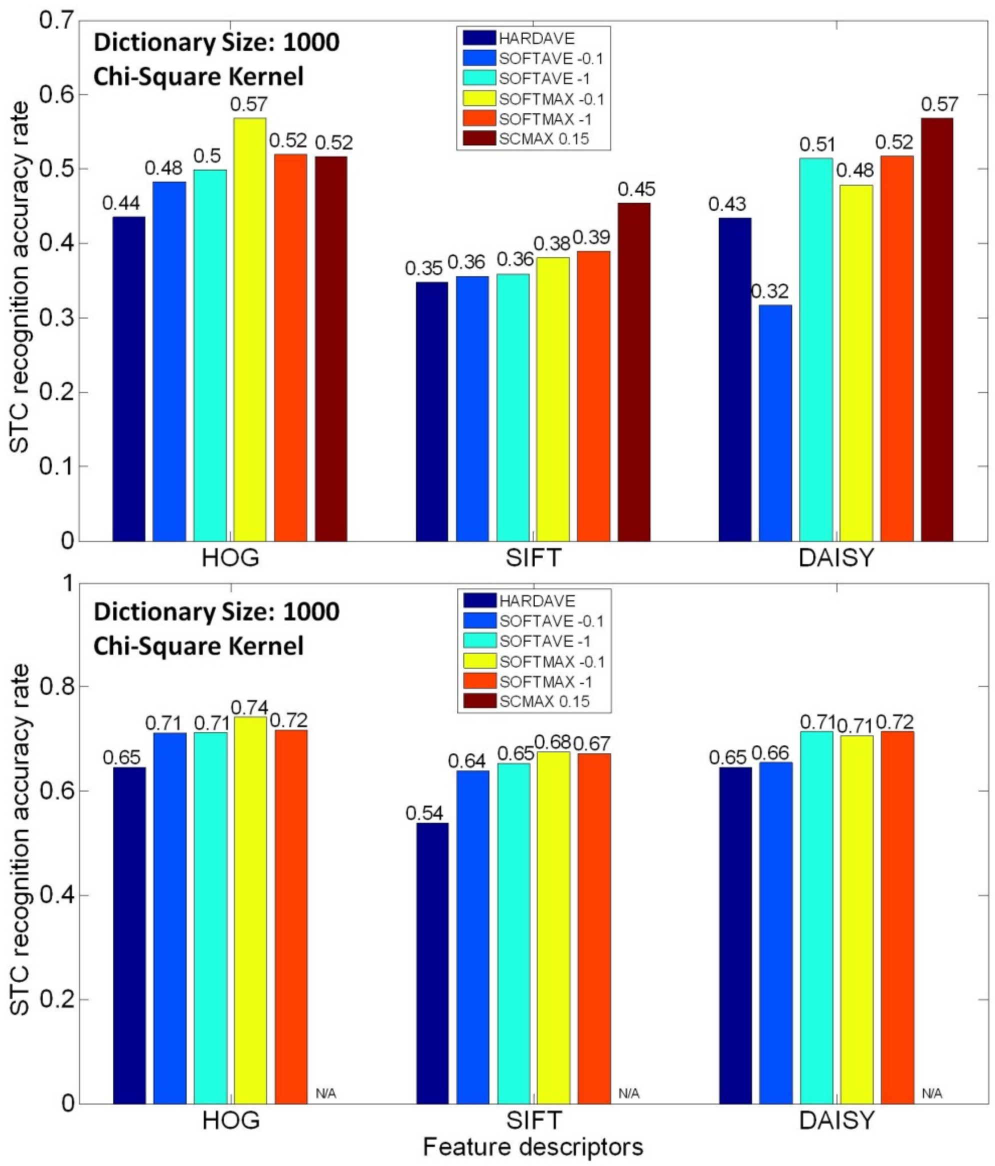


Fig. 8–5. Evaluating 6 coding/pooling schemes, under three feature descriptors, dictionary size 1000, and Chi-Square SVM kernel. Top figure is from CHARS74K and bottom figure is from ICDAR2003CH.

Above experimental results show that SOFT coding and MAX pooling obtains the best performance in the two datasets. Now we compare the performance of SOFTMAX with Fisher vector, still on the basis of HOG, SIFT and DAISY descriptors.

The experimental results are shown in Fig. 8–6. It demonstrates that Fisher vector obtains even better performance than SOFTMAX and all other coding/pooling schemes in BOW model.

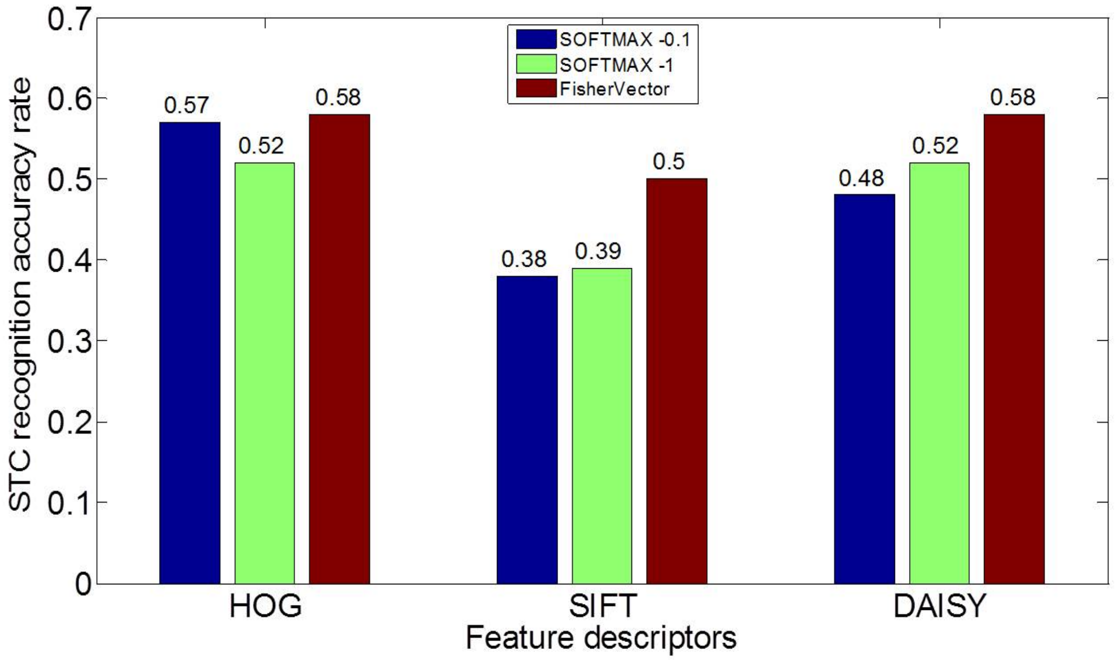


Fig. 8–6. STC prediction results in Chars74K, including all kinds of coding and pooling schemes. Fisher vector obtains better performance than the others.

8.2.4. Evaluating SVM Kernel

Besides the design of STC feature representation, the choice of classification model plays an important role in STC prediction. The feature vector of a character patch, which is a histogram of visual words, is regarded as an observation point in classification model. Currently, all the experimental results of STC prediction are obtained from SVM learning models with linear kernel and χ^2 kernel [**87**].

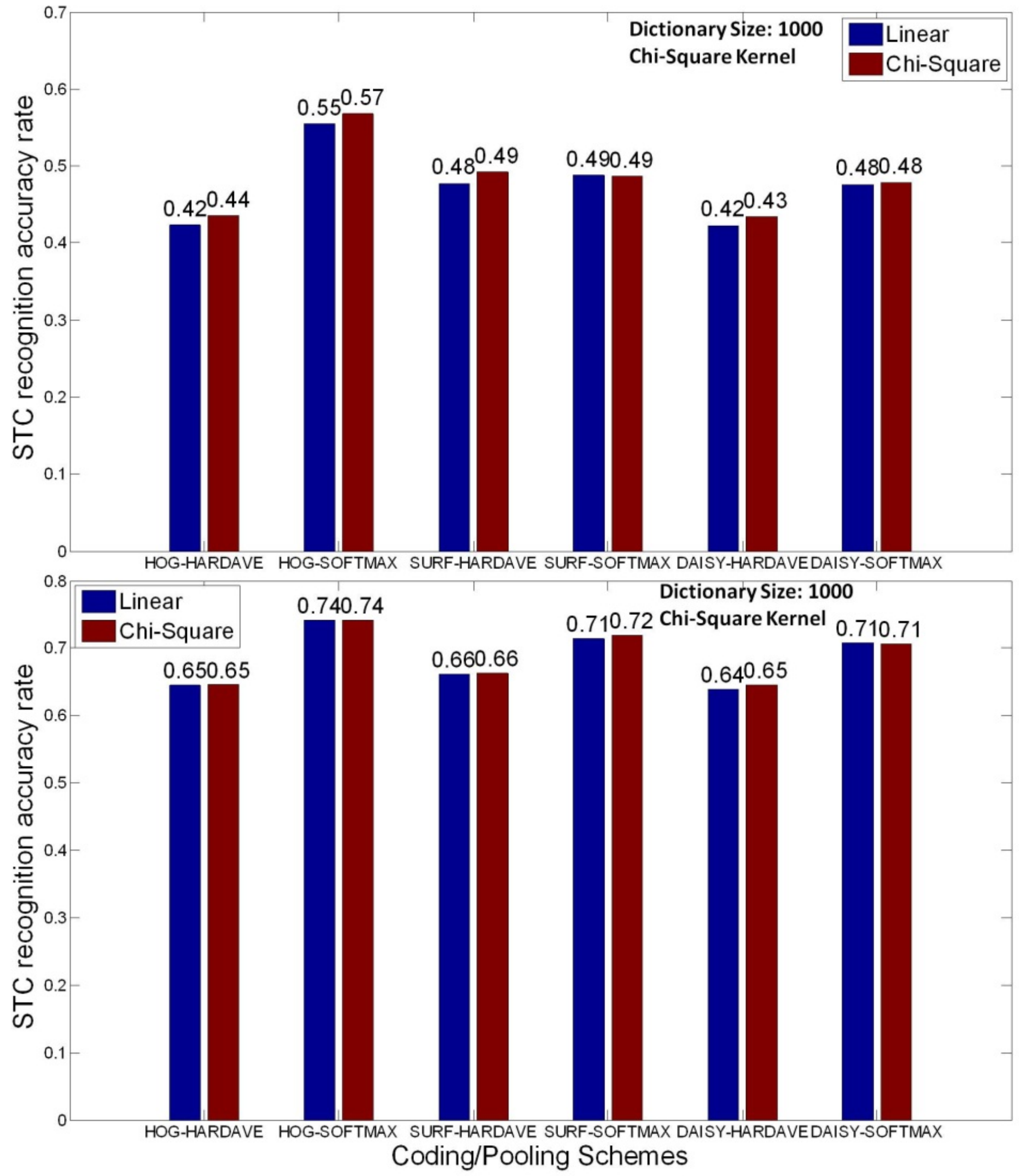


Fig. 8–7. Evaluating linear kernel and χ^2 kernel in SVM. Top figure is from CHARS74K and bottom figure is from ICDAR2003CH.

8.3. Evaluating Feature Representations of Global Sampling

8.3.1. Complete STC in Global Sampling

Above experiments are based on densely local sampling of key-points in character patch. Feature descriptor is computed from each key point and then normalized and pooled through coding-pooling schemes. In this section, we propose a new feature representation for STC prediction based on global sampling. We extract GHOG features directly from the whole character patch. Compared to local sampling, the GHOG based global sampling obtains even better performance of STC prediction. Because global sampling has the following advantages: (1) there is no coding so no information loss (2) spatial structure is preserved when concatenating descriptors of grids in order. The evaluation results show that GHOG obtains accurate rate up to 0.62 at CHARS74K and 0.76 at ICDAR2003, which are better than the highest results (0.58 at CHARS74K and 0.75 at ICDAR2003) in local sampling. In addition, GHOG outperforms most existing methods as shown in Table 8–1.

8.3.2. Incomplete STC in Global Sampling

STC prediction is usually based on the resulting character patches from text region detection and STC segmentation. However, the two steps cannot ensure complete character patches. We evaluate the performance of GHOG on these incomplete (truncated) character patches and illustrate the results in Fig. 8–8. It shows that more complete structure obtains better recognition performance, and the top and bottom parts of character patch generate more discriminative structure features than the left and right parts.

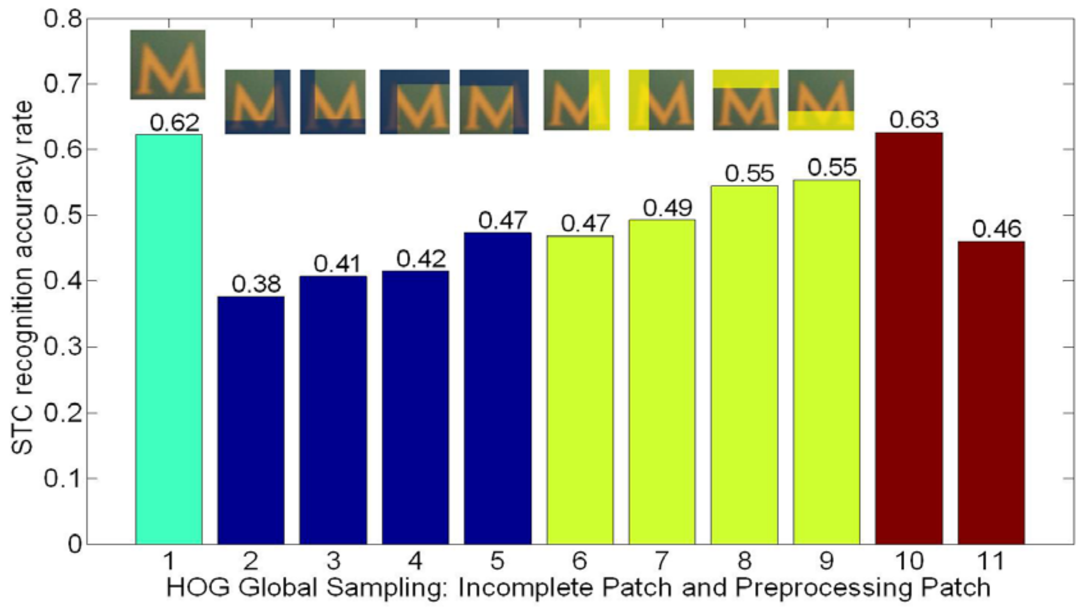


Fig. 8–8. Performance evaluations based on GHOG global sampling. The left first bar denotes the accurate rate of global HOG in complete and original patch. Blue bars and yellow bars denote accuracy rate of incomplete patches as their top examples. The first red bar denotes accuracy rate (0.63) of preprocessed patches by Gaussian and the second red bar denotes that of Laplacian filters (0.46).

Moreover, we apply Gaussian and Laplacian filters to preprocess character patches before extracting GHOG features. Gaussian filter removes background noise, while Laplacian filter emphasizes the boundary that contains much information on character structure. The evaluation results (see Fig. 8–8) show that Gaussian smooth improves the recognition performance slightly, but Laplacian lowers the performance because the noise negatively influences HOG descriptors.

Table 8–1. Comparations between our best results and existing methods of STC prediction over the benchmark datasets.

	CHARS74K-15	ICDAR2003CH
Global HOG+SVM	**0.62**	**0.76**
Local HOG+SVM	**0.58**	**0.75**
Geometrical blur + NN [**19**]	0.47	/
Geometrical blur + SVM [**19**]	0.53	/
Shape context + NN [**19**]	0.34	/
Shape context + SVM [**19**]	0.35	/
Multiple kernel learning [**19**]	0.55	/
ABBYY [**19**]	0.31	/
Coates method [**16**]	/	0.82
HOG+NN [**90**]	0.58	0.52
SYNTH+FERNS [**90**]	0.47	0.52
NATIVE+FERNS [**90**]	0.54	0.64

NN: Nearest neighbor classification; SYNTH: synchronic patch for training; NATIVE: native scene image patch for training

8.4. Datasets and Evaluation Measures of Word Recognition

ICDAR-2011 robust reading dataset contains 484 scene images in total, in which 229 images with 619 ground truth text regions are used for training and 255 images with 934 ground truth text regions are used for testing in ICDAR 2011 Robust

Reading competition. The image size ranges from 422×102 to 3888×2592. The proposed framework is applied to the text regions for word recognition.

The performance of word recognition is evaluated based on edit distance between ground-truth words and recognized words. Whenever a word is updated by inserting a new character, deleting a character or modifying a character, its edit distance from the original word is increased by 1. However, it might be too strict to only compute the accuracy of perfectly recognized words with edit distance 0 (ED0) to ground truth. We also measure word recognition of the edit distance 1 (ED1) to ground truth. This is because even though the recognized word has 1 edit distance from, we still can recognize it in most cases.

8.5. EVALUATING WORD RECOGNITION

Two experiments are carried out to evaluate the performance of word recognition on the basis of STC prediction. In the first experiment, we perform word recognition by sequentially combining the results of STC predictions in a text region, but not adopt CRF model to correct the STC predictions. In the second experiment, CRF model is adopted to combine vision-based STC prediction with lexicon-based word configuration knowledge.

Table 8–2. The experimental results of word recognition in ICDAR 2011, which are evaluated by ED0 and ED1.

	Sequential STC prediction	CRF correction
ED0	0.161	0.210
ED1	0.432	0.433

From the experimental results, we can observe that word recognition is a challenging task. A tolerance of one edit distance will much improve the performance. Besides, the CRF model is able to improve the performance of word recognition, but the improvement is not significantly large. It shows that word recognition mostly relies on the accuracy of STC prediction based on STC feature representation.

8.6. SUMMARY OF CONTRIBUTIONS

In this chapter, the proposed methods of scene text character recognition are evaluated over benchmark datasets, in which a comparative performance evaluation of different low-level descriptors, dictionary sizes and coding/pooling schemes is carried out. The experimental results demonstrate the effectiveness of our method in recognizing the 62 categories of scene text character. From the perspective of quantitative analysis, our method outperforms the state-of-the-art results in the benchmark datasets.

Chapter 9 BLIND-ASSISTANT HAND-HELD OBJECT RECOGNITION

9.1. BLIND-ASSISTANT PROTOTYPE SYSTEM

Scene text extraction can be widely used in blind assistance. The 2008 National Health Interview Survey (NHIS) [**58**] indicates that about 25.2 million adult Americans (8% of the total population) are visually impaired. There are about 314 million visually impaired people all over the world, and 45 million of them are blind [**94**]. A number of assistant reading systems have been designed specifically for the blind or visually impaired people [**41**] [**54**] [**81**] [**73**].

Many blind assistant systems are developed to help visually impaired people through some wearable devices [**17**]. For example, a portable bar code reader is designed to help blind people identify different products in an extensive product database, and it enables users who are blind to access information about these products [**71**] through speech and braille. But a big limitation is that it is very hard for blind users to find the position of the bar code and to correctly point the bar code reader at the bar code. To our knowledge, no existing system can read text from the kinds of challenging patterns and backgrounds found on many everyday commercial products. To assist blind or visually impaired people to read text from these kinds of hand-held objects, we have conceived of a camera-based text reading scheme to track the object of interest within the camera view and extract print text information from the object. The system demonstrates that our framework of scene text extraction can effectively handle complex background and multiple patterns, and obtain text information from both hand-held objects and nearby signage, as shown in Fig. 9–1.

In most assistive reading systems, users have to position the object of interest within the center of the camera's view. According to our survey, there are still no acceptable solutions. We try to handle this problem step-by-step. To make sure the hand-held object can be easily captured in the camera view, we use a camera with sufficiently wide angle to accommodate users with only approximate aim. However, this wide angle camera will also capture many other text objects (for example while shopping at a supermarket). To centralize the hand-held object from the camera image, a motion-based method is adopted to acquire a region of interest (ROI) of the object. Then we perform scene text extraction from only this ROI, including detecting text regions and recognizing text codes. In the end, the recognized text codes are output to blind users by audio device. To present how our prototype system works, a flowchart is presented in Fig. 9–2.

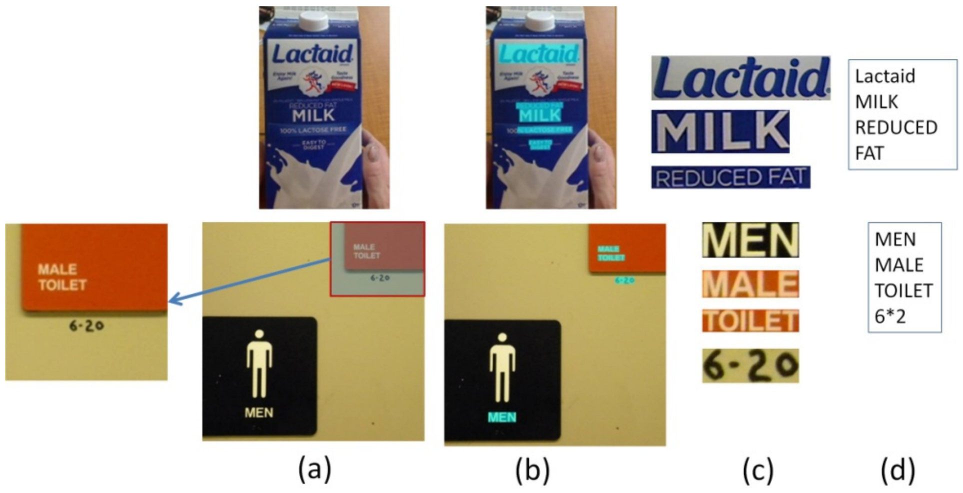


Fig. 9–1. Two examples of extracting text by the prototype system from camera-captured images. Top: a milk box; Bottom: a men bathroom signage. (a) camera-captured images; (b) localized text regions (marked in blue); (c) text regions cropped from image; (d) text codes recognized by OCR. The top-right portion of the bottom image that contains text is also shown in a magnified callout, for clarity.

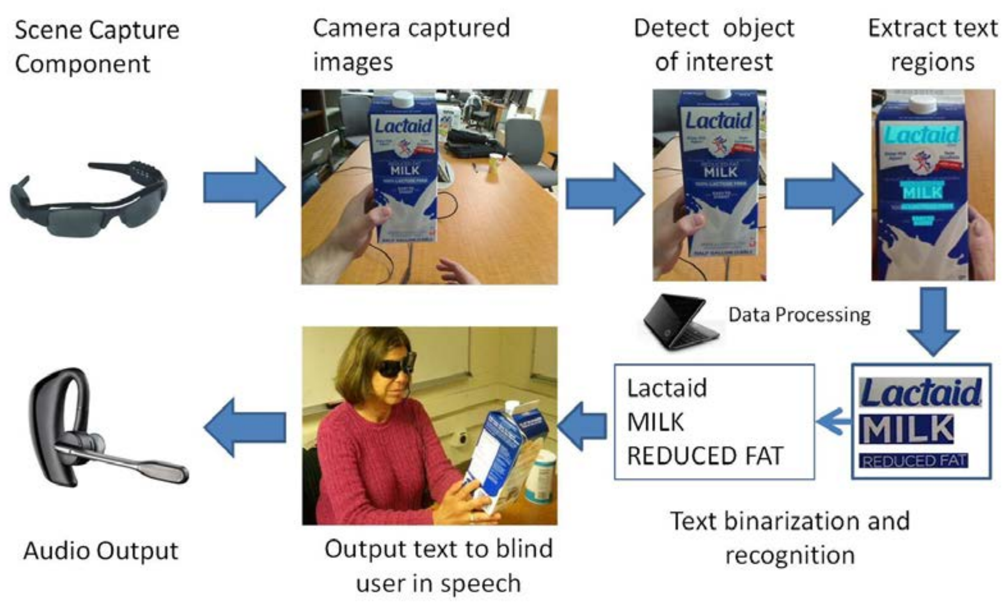


Fig. 9–2. Flowchart of our prototype system to read text from hand-held objects for blind users.

9.2. IMPLEMENTING SCENE TEXT EXTRACTION IN PC PLATFORM

A prototype system of scene text extraction is designed and implemented in PC platform, which is compatible with Windows and Ubuntu-linux platforms. Fig. 9–3 illustrates the hardware of this prototype system based on our proposed framework. This system consists of three components: scene capture, data processing and audio output. The scene capture component collects surrounding scenes or objects containing text, and the captured data is in the form of images or video. In our current prototype system, this component is implemented by a camera attached to a pair of sunglasses. The data processing component is used for deploying our proposed framework. In our current prototype system, a min-laptop is used as the processing device in our current prototype system. The audio output component is to inform the blind user of recognized text codes. A Bluetooth earpiece with mini-microphone is adopted for speech output. This simple hardware configuration proves the portability of the assistive text reading system.

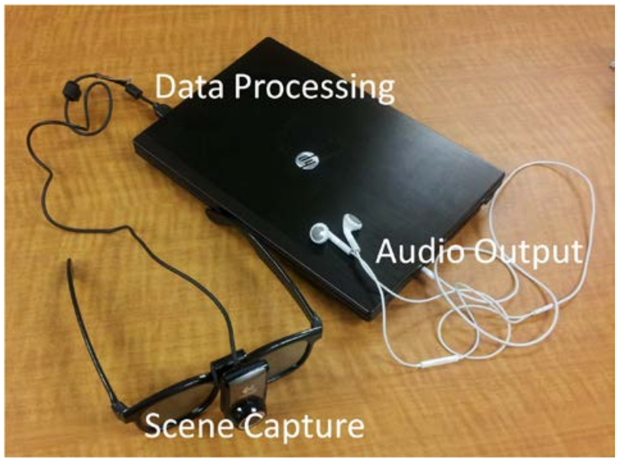


Fig. 9–3. A snapshot of our prototype system, including three functional components for scene capture, data processing and audio output.

The core component of this prototype system is data processing, which is implemented according to the proposed scene text extraction framework in this dissertation. It consists of 5 modules, extracting candidate character components as Chapter 2, extracting candidate string fragments as Chapter 3, modeling structural feature representations as Sections 4.1-4.3 and Section 5.3, learning the classifier of true positive text fragment sample as Section 4.4, learning the classifier of STC prediction as Section 5.5 (LEARNING), data I/O and management (DATA), and system testing module (TESTING). The whole system is implemented by C++ programming language. In our implementations, OpenCV 2.4 library [**64**] is employed to support some basic data structure like Matrix, and some basic algorithms related to low-level image processing like Canny Edge Detection and feature descriptors like SIFT and HOG. LIBSVM [**85**] is employed to implement SVM training and testing processes. VLFeat [**86**] library is employed to implement MSER operator which is used to extract possible character component.

The prototype system has been used to assist blind or visually impaired people to recognize hand-held object as described, as shown in Fig. 9–4.

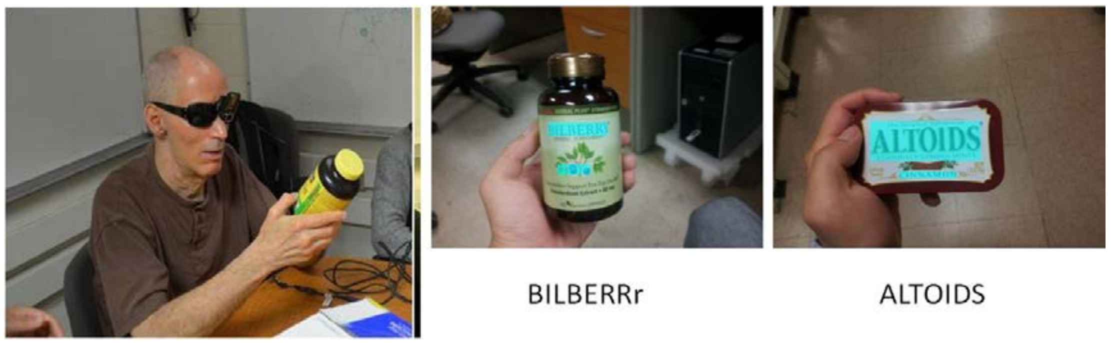


Fig. 9–4. Prototype system assists blind user read text information from hand-held objects, including the detected text regions in cyan and the recognized text codes.

Currently, the system efficiency is decided by the efficiency of scene text extraction in each image or video frame. But we will design parallel processing of text extraction and device input/output to further improve the efficiency of this assistant reading system. That is, speech output of recognized text and localization of text regions in the next image are performed simultaneously.

9.3. Evaluation Results and Discussions

To evaluate the performance of the prototype system and develop a user-friendly interface, following Human Subjects Institutional Review Board approval, we recruited 10 blind persons to collect a dataset of reading text on hand-held objects. The hardware of the prototype system includes a Logitech web camera with autofocus, which is secured to the nose bridge of a pair of sunglasses. The camera is connected to a mini laptop by a USB connection. The laptop performs the processing and provides audio output. To avoid serious blocking or aural distraction, we will choose a wireless "open" style Bluetooth earpiece for presenting detection results as speech outputs to the blind users in a full prototype implementation.

The blind user wore the camera/sunglasses to capture the image of the objects in his/her hand, as illustrated in Fig. 9–5. The resolution of the captured image is 960×720. There were 14 testing objects for each person, including grocery boxes, medicine bottles, books, etc. They were required keep their head (where the camera is fixed) stationary for a few seconds and subsequently shake the object for an additional couple of seconds to detect the region of object of interest. Each object was then rotated by the user several times to ensure that surfaces with text captions are exposed and captured. We manually extracted 116 captured images and labeled 312 text regions of main titles.

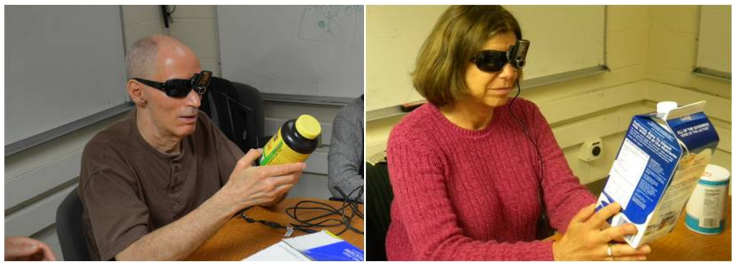


Fig. 9–5. Examples of blind persons are capturing images of the object in their hands.

The user-captured dataset of object text is used to evaluate our prototype system. In our evaluations, a region is correctly detected if the ratio of the overlapping area of a detected region and its ground truth region is no less than 3/4. Experiments show that 225 of the 312 ground truth text regions are hit by our localization algorithm. By using the same evaluation measures as above experiments of scene text detection, we obtain precision ***0.52***, recall ***0.62***, and *f*-measure ***0.52*** over this dataset. The precision is relatively lower than that on the ICDAR Robust Reading Dataset. It is because the images in the blind-captured dataset have lower resolutions and more compact distribution of text information, so they generate low-quality edge maps and text boundaries, which result in improper spatial layouts and text structural features. Some examples of extracted scene text from hand-held objects are illustrated in Fig. 9–6, proving that our proposed framework is suitable for real applications.

In this application, off-the-shelf OCR is adopted for scene text recognition in the detected text regions. Since the existence of false positive text regions and the low recognition accuracy of OCR on scene text regions, the practical system would restrict the range of possible recognized words by a prior dictionary of common words that are frequently printed in hand-held objects. A text extraction result is output by audio only if it has close edit distance to some word in the dictionary.

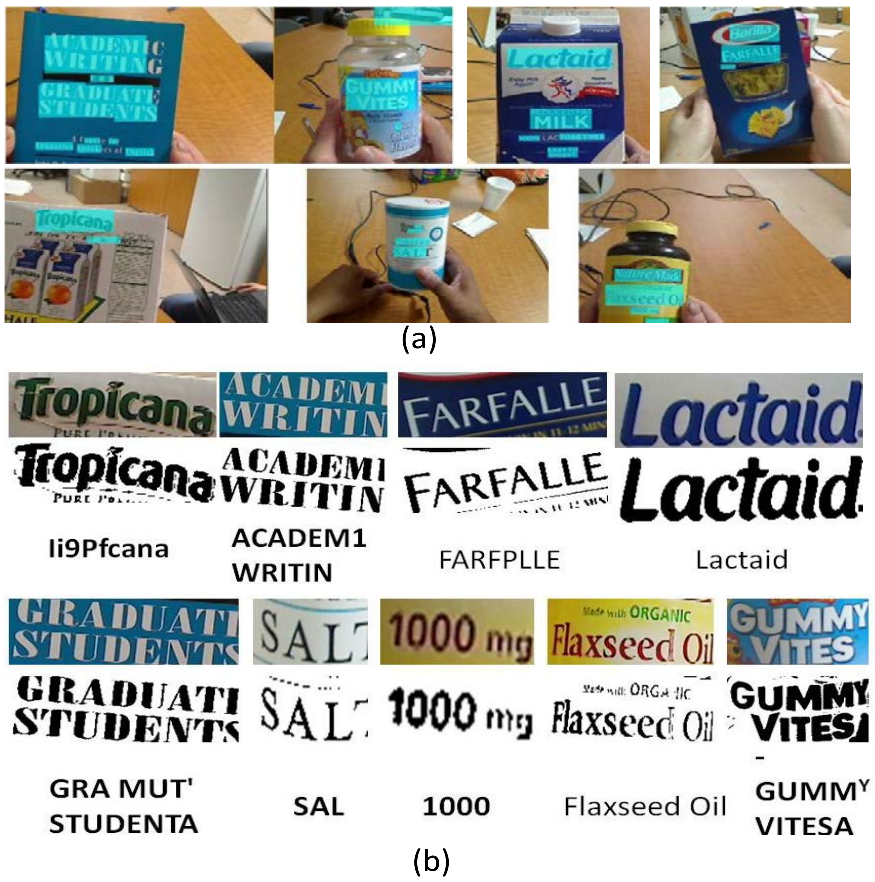


Fig. 9–6. (a) Some results of text detection on the user-captured dataset, where localized text regions are marked in blue. (b) Two groups of enlarged text regions, binarized text regions, and word recognition results from top to down.

9.4. SUMMARY

In this chapter, we combine scene text extraction in PC platform and combining it with background subtraction algorithm to build a blind assistant system of hand-held object recognition.

Chapter 10 TRACKING-BASED SCENE TEXT RECOGNITION

Due to the cluttered background and multiple text patterns, scene text recognition even from a detected text region is still a challenging problem. Most OCR systems are designed for scanned documents with relatively clean background or segmented STCs, and they do not obtain good performance over scene text as mentioned in Chapter 9.

On the other hand, in a number of real-world applications of text information retrieval, the raw data captured from natural scene is in the form of video frames rather than a single scene image. This means frame relationships are ignored in the traditional scene text recognition methods as described in Chapter 5 based on a single image. Thus we propose a method of scene text recognition from a video of natural scene. In our method, a text recognition method can be combined with a tracking algorithm to improve the recognition accuracy. We carry out text detection on the first frame of a given video, and generate an initial bounding box of text region. Then we adopt an object tracking algorithm to the detected text region and obtain its bounding box in succeeding frames. Next, STC prediction and CRF model are applied to the tracking boxes of text regions to recognize text information.

10.1. SCENE TEXT TRACKING

At first, scene text detection based on MSER detector and adjacent character grouping is performed to extract bounding boxes of text regions from the first video frame. In this process, we can obtain bounding box of each STC, which will serve as the input to scene text tracking. Text tracking will generate corresponding STC bounding boxes in succeeding video frames, as shown in Fig. 10–1.

To simultaneously track several STCs belonging to the same word, multi-object tracking is applied to scene text regions. In scene text scenario, we can avoid some challenges of multi-object tracking by the three constraints. First, we do not need to estimate the STC trajectories in the same word independently because we can instead estimate the trajectory of the whole word at first as a hint. Second, the STCs in the same word are well aligned and have relatively low dependencies with each other. Third, the locations of characters are always stable. So the inter-object occlusions rarely happen as long as the whole word is clearly captured.

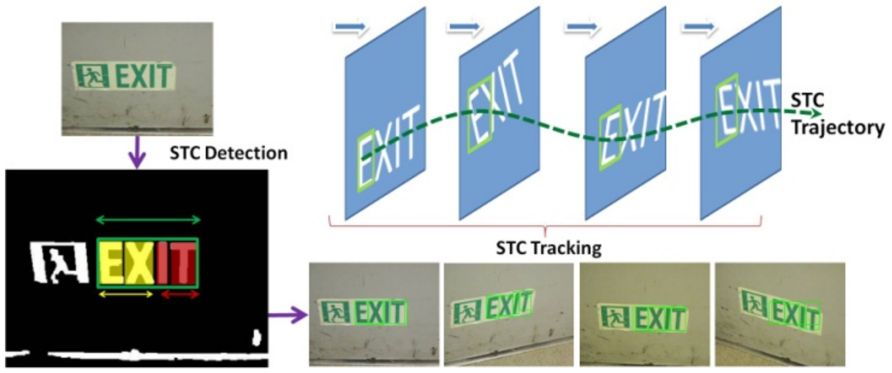


Fig. 10–1. Initial text regions are extracted by grouping adjacent connected components in similar size and horizontal alignment in the MSER map. Each scene text character will be independently tracked with the multi-object tracking method. A trajectory is then estimated by merging the tracked STC bounding box in each frame. The STC prediction scores along this STC trajectory will be used to improve scene text recognition.

We adopt the tracking by detection method in our framework, i.e., each STC is traded as an independent object model that is detected and tracked in continuous multiple frames [**63**] [**96**]. An online tracking model in [**27**] is able to handle the variations of lighting and appearance. Compared with other tracking methods such as the first-order Markov chain model which predicts object location in next frame from that in current frame, these tracking by detection method category successfully solves the re-initialization problem even when a target has been lost in some frames accidentally and the excessive model drift problem due to similar appearances of some STCs.

At each frame in tracking process, the algorithm searches for the globally optimal hypothesis for the location of each STC. The detection output is then used as a constraint to optimize the trajectory search. Optimized trajectory estimation is then fed back to guide text detection in subsequent frames and reduce the effects of motion blur. The two processes are iteratively performed to track STC bounding boxes.

10.2. STC Prediction and Word-Level Configuration

A tracked STC bounding box is cropped from video frame of natural scene, and STC prediction as described in Chapter 5 is performed to label the most probable category of the STC from the 62 candidate categories. A 62-dimensional prediction score is obtained from STC prediction to represent the probability of each category.

Based on the prediction labels and scores, we can infer the involved text information by using three schemes as described in Section 10.4. First, we could just randomly select a tracking frame and use its STC prediction results as the final results. Secondly, we could use majority voting among all video frames to decide the final results of STC predictions. Third, we could extend the CRF model as described in Section 6.2 to fuse the prediction results in all video frames in an optimized way.

10.3. Dataset and Evaluation Measures

Since our framework works on multiple frames, we first collect a video dataset of text information from natural scene, which consists of 50 videos including both indoor and outdoor environments. These videos are captured by a moving and shaking camera, which results in some motion blur. Each video sample in this dataset contains more than 300 frames, where the target signage is shot from about $-45^{\circ} \sim 45^{\circ}$ view angles. The difference between neighboring frames is not quite large, so we uniformly sample 100 frames from each video as the effective frames in our experiments. The following experiments extract the STC trajectories in the 100 effective frames to perform multi-frame scene text recognition.

To evaluate the performance of scene text recognition, we use different measurements for STC prediction and word recognition. In STC prediction, we measure the accuracy in a testing set by CH-RATE. In word recognition, the difference between two words is measured by the edit distance. Whenever a word is updated by inserting a new character, deleting a character or modifying a character, its edit distance from the original word is increased by 1. However, it might be too strict to only compute the accuracy of perfectly recognized words with edit distance 0 (ED0) to ground truth. We also measure word recognition of the edit distance 1 (ED1) to ground truth. This is because even though the recognized word has 1 edit distance from, we still can recognize it in most cases.

10.4. EXPERIMENTAL RESULTS AND DISCUSSIONS

To validate the effectiveness of multiple frames in scene text recognition, we first carry out the experiment by using a single frame. Each sample in our dataset consists of multiple frames, and we could randomly choose one frame and perform scene text recognition. However, the image quality of the video frames is largely different, so it is unreasonable to use only one frame to evaluate text recognition in a sample. In our experiments, 10 frames, i.e., $1/10$ of the total number of effective frames in a video, are randomly selected to evaluate scene text recognition. Each STC bounding box generates a vector of prediction scores in 62 dimensions, since we define 62 categories of STCs. We compute the prediction scores of the corresponding STCs in the same trajectory from the randomly selected frames, and then calculate their mean as the result of scene text recognition in a single frame.

Next, we evaluate multi-frame text recognition. In text tracking, STC bounding boxes obtained from text detection are extended into succeeding frames. It is worth noticing that although the motion blur due to camera shaking is well handled, not all

STC bounding boxes in video frames are correctly tracked, and not all STC categories are correctly predicted. Therefore, we fuse the STC prediction scores in each frame along its trajectory to improve the recognition accuracy. We propose two fusion methods. The first fusion method employs Majority Voting model, which makes statistics of category labels of STC prediction in all frames and choose the one in the highest frequency as the final result. For each STC trajectory, we generate a vector of predicted category labels from the frames. Then the highest-frequent label is computed as the result of STC prediction. All STC prediction results are then cascaded into word recognition result.

The second fusion method employs CRF model to fuse multi-frame STC prediction scores under the lexical constraints. CRF model is learned from ground truth text regions in CHARS74K, ICDAR2003, and ICDAR2011 datasets. We perform scene text detection and STC prediction in those ground truth regions, and use the prediction scores and bigram lexicons to train CRF model. Then the CRF model is applied to the STC prediction labels and scores in multiple frames of text tracking. Fig. 10–2 demonstrates some example results of text tracking and word recognition from multiple frames.

Table 10–1. The results of word recognition in a single frame and multiple frames of text racking.

	CH-RATE	ED0	ED1
Single Frame	0.640	0.333	0.583
Multi-Frame (Majority)	0.680	0.389	0.611
Multi-Frame (CRF)	0.713	0.389	0.722

The experimental results in Table 10–1 demonstrate that multi-frame scene text recognition significantly improves the performance of STC prediction and word recognition in comparison with single-frame recognition. Majority voting suppresses the minority frames that generate incorrect STC prediction, so it obtains better performance of STC prediction and word recognition than single-frame method. Furthermore, CRF brings in lexical prior knowledge of bigram STCs. It further improves the performance of STC prediction and word recognition. Since the bigram lexical prior is calculated from ground truth words of three other datasets, and the size of our self-collected text video dataset is not large enough, CRF obtains the same performance on ED0 as majority voting. But we infer that CRF will give better performance as the size increasing of text video dataset.

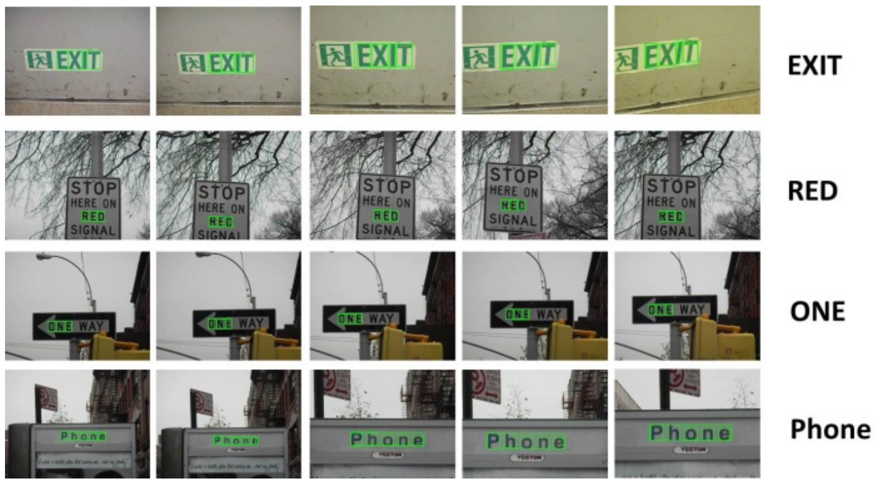


Fig. 10–2. Some example results of word recognition from multiple frames of text tracking. Right column shows recognized words.

10.5. Summary

In this chapter, we combine scene text extraction with text tracking algorithm to design an algorithm of scene text recognition in multiple video frames.

Chapter 11 LICENSE PLATE DETECTION

Another application of scene text extraction is to detect license plate (LP) in a car, which is captured from rear view. LP detection plays an important role in many applications such as car identification and privacy protection. Based on scene text detection, we design an LP detection algorithm to automatically find out LP regions in scene image. It is able to handle multiple LPs in different sizes and positions. The LP algorithm consists of three components, car detection, scene text detection and edge-based post processing. It extracts the possible LP regions step-by-step from complex background.

11.1. CAR DETECTION

Car detection is to extract car regions from a scene image, for reducing LP search domains. Car instances in our experiments are all captured from backside, so we train a car-rear classifier through Haar features and Adaboost learning model as [**88**]. Some samples from car-rear dataset are shown in Fig. 11–1.

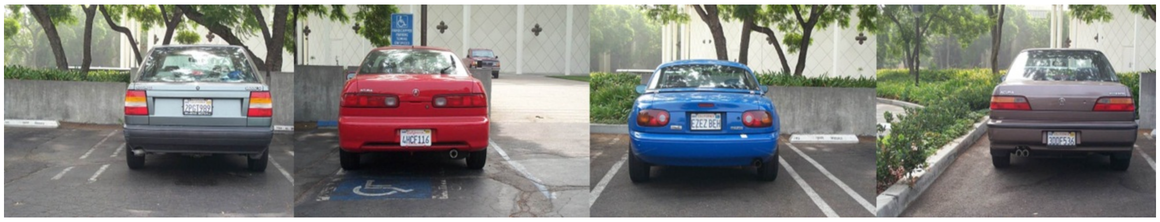


Fig. 11–1. Some examples of car-rear training samples. Each car is located at complex background. The LPs are mostly located in the central parts of car rear.

Next, sliding window is employed to search the whole scene image for possible car regions by using the car-rear classifier. As shown in Fig. 11–2(c), red box depicts a detected car region.

11.2. LICENSE PLATE LOCALIZATION

To localize the LP within a detected car region, we should extract the sub regions with text information, and filter out those with tires and bumper. Scene text detection algorithm is employed to extract text characters and strings from car region. This algorithm can be divided into two steps, rule-based layout analysis and learning-based structural feature analysis, as respectively described in Chapter 2, Chapter 3 and Chapter 4. The rule-based analysis assumes that text information in scene image consists of text characters in similar size and horizontal alignment. At first, connected components of possible text characters are extracted from boundary extraction in edge map of scene image. Then a group of constraints is defined to filter out connected components from background outliers, and these constraints are related to size, aspect ratio and the number of inner holes. Next, we perform adjacent character grouping as described in Section 3.3 to extract the candidate components aligned with neighbors, and combine these neighboring components into a candidate patch. A text classifier is then applied to the candidate patch to predict whether it contains text or not. This text classifier is obtained by learning-based text feature analysis. First we build a training set by using text patches as positive training samples and non-text background patches as negative training samples. From each training sample, we extract text features based on gradient distributions and stroke orientations and normalize them into feature vector. Then cascaded adaboost model is applied to train a text classifier by all feature vectors from training set. In the end, the neighboring text patches in similar sizes and heights are merged into text region, as the cyan rectangle regions in Fig. 11–2(c).

11.3. POST PROCESSING

Furthermore, a post process based on edge density is designed to filter out false positive text regions. An LP is usually located at the central part of a car region, and it has much higher edge density than all the other backside car components including bumper, car-body, windshield, and taillight, as shown in Fig. 11–2(b). Based on these observations, two conditions are defined to crop out valid LP region from the detected text regions. Firstly, for a rectangle text region and a rectangle car region, the text region will be removed if one of the two conditions is satisfied; 1) the height of the text region is larger than 1/4 height of the car region; 2) the distance between their left-sides (or right-sides) is smaller than 1/5 width of the car region, or the distance between their top-sides is smaller than 1/5 height of the car region. Second, the edge density of each text region is calculated as the number of edge pixels divided by the area of the region, and we preserve the region with highest edge density as detected LP region (see Fig. 11–2(d) and Fig. 11–3).

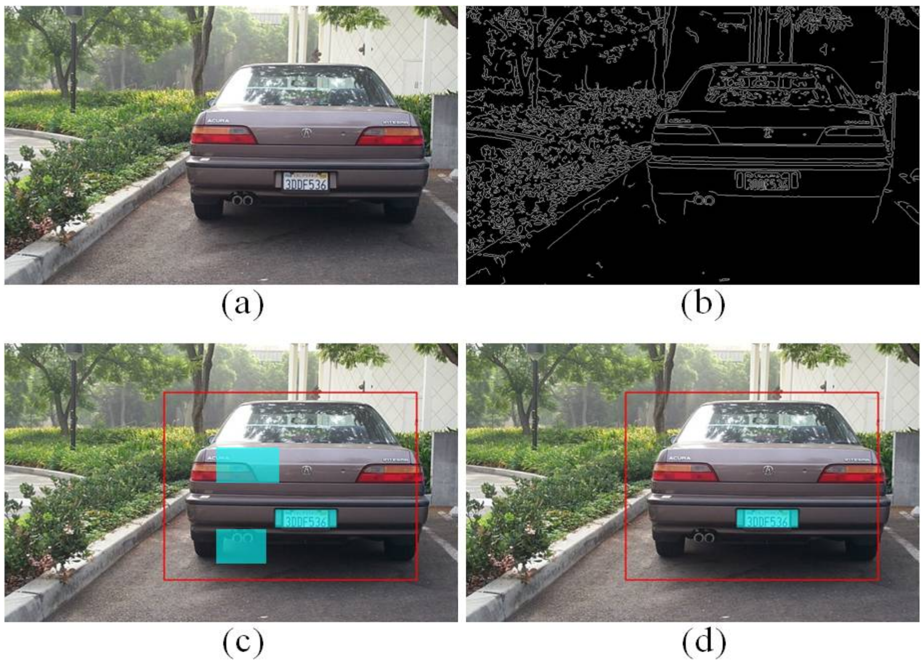


Fig. 11–2. (a) Original image of license plate. (b) Edge map from canny edge detection. (c) Detected car region in red box and detected text regions in cyan rectangle boxes. (d) Post-processing is applied to filter out background outliers and preserve the region with the highest edge density as ultimate LP region.

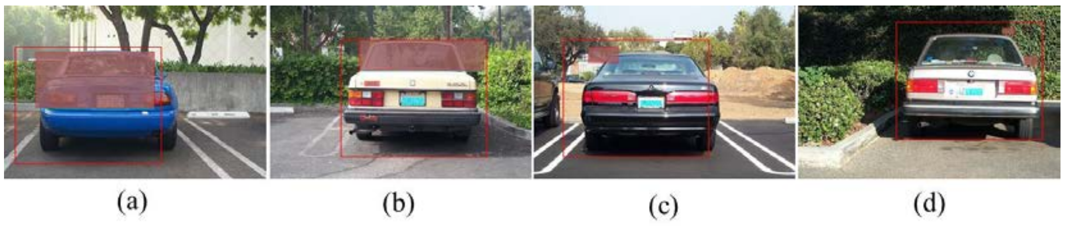


Fig. 11–3. Some post-processing examples in LP detection. Cyan regions inside red car boxes represent the true positive LP, and red regions inside red car boxes represent false positive LP detections that are filtered out by post processing.

11.4. Evaluation Results and Discussions

To evaluate the performance of LP detection, we adopt a dataset [**10**] of car rears with LPs in high enough resolutions. This dataset contains 126 natural scene images captured from complex background. The experimental results show that our framework is able to detect 74 of the 126 LPs completely, as shown in Fig. 11–2(d) and Fig. 11–3. Besides, 46 of the 126 LPs are partially detected, as shown in Fig. 11–4(a). The remaining 6 LPs are missing because of severe illumination or low resolution, as shown in Fig. 11–4(b).

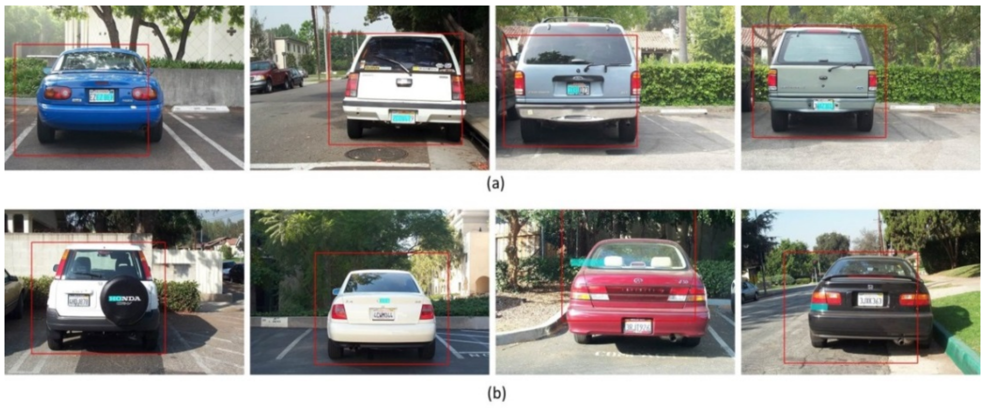


Fig. 11–4. (a) LPs are partially detected. (b) Some LPs are missing because of severe illuminations or low resolutions.

11.5. Summary

In this chapter, we combine scene text extraction with car detection to localize license plate in a natural scene image.

Chapter 12 INDOOR NAVIGATION

A blind-assistant prototype system is designed for hand-held object recognition in Chapter 9. We can further extend the system to indoor navigation, by extracting indicative information from surrounding signage in indoor environment. In most cases, indoor navigation is to lead blind users to a specific office, a restroom or an elevator entrance. All of them have doors by a room name or a room number. The people in normal vision can refer floor plan map to find their ways, but blind or visually impaired people cannot acquire this information. Thus our proposed prototype system can perceive their current location and generate a proper path from current location to their destination.

The hardware of this prototype system is similar to the system of hand-held object recognition in Chapter 9, including wearable camera, process unit, and audio output device. However, the system implements indoor navigation by adding in door detection and floor plan parse algorithms, which will be described in detail.

12.1. DOOR DETECTION AND ROOM NUMBER EXTRACTION

In both floor plan map and real indoor environment, doors serve as important landmarks and transition points for way-finding. They also provide entrance and exit information. Thus, an effective door detection method plays an important role in indoor navigation. Our system adopts the vision-based door detection method presented in [**97**] to localize surrounding doors for blind users.

This method adopts a very general geometric door model, describing the general and stable features of doors—edges and corners, as shown in Fig. 12–1. This method can handle complex background objects and remove most door-like shapes.

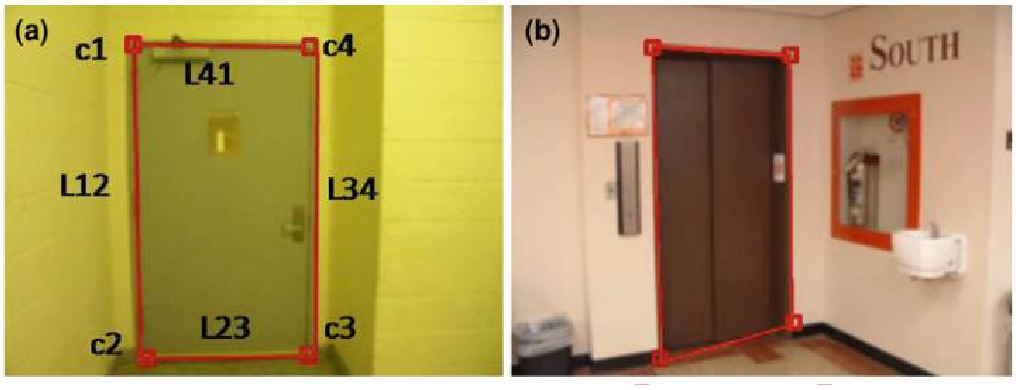


Fig. 12–1. (a) Edges and corners are used for door detection. (b) Door detection under cluttered background.

Based on the detected doors, scene text extraction is performed within the door region or around its immediate neighboring region, for obtaining text information related to room names and room numbers, as shown in Fig. 12–2.

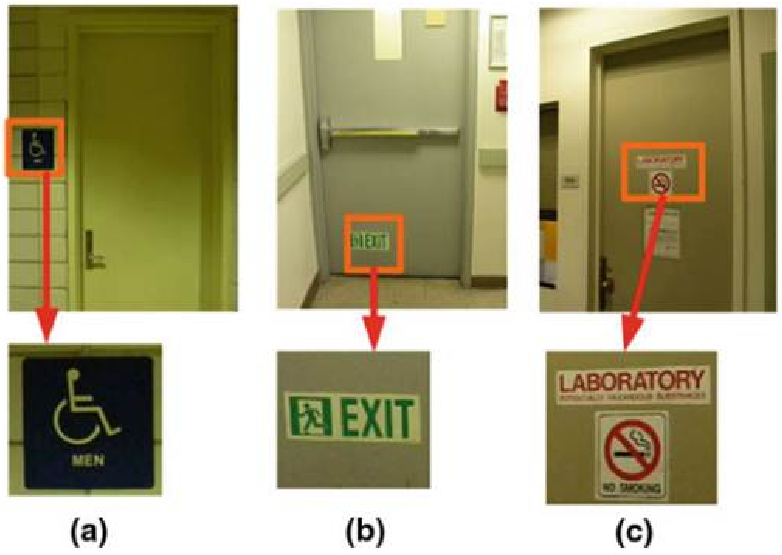


Fig. 12–2. Door detection and further scene text extraction from doors. This is a combination of door detection and scene text extraction techniques for generating current locations.

The doors and its attached text information serve as a valuable indicator for navigation. In general, it is difficult for blind user to take high-quality images or videos [**82**] for information retrieval. We design a method to select high-quality image frame from a camera-based video. Also many sophisticated algorithms were proposed [**44**] [**24**] [**8**] [**14**] to perform de-blurring and de-noising in camera-based scene image, which will be integrated into this system.

12.2. Floor Plan Parse for Way-finding

Most buildings custom floor plan maps as tourist guide, as shown in Fig. 12–3(a). A floor plan map contains room numbers and relative locations of the offices, restrooms, and elevator entrances in this building. However, floor plan map can only provide static information without tracking the locations of blind users. Thus, our navigation system combines floor plan parse and vision-based information retrieval techniques such as door detection and scene text extraction to figure out a solution of blind-assistant way-finding in unfamiliar buildings.

A primary design of floor plan based way-finding can be found in [**35**]. Our design is as follows. A floor plan map will be parsed into a graph, in which a room is defined as a node (see Fig. 12–3(b)). Each pair of nodes is connected by an edge, and an available path of way-finding is defined on each edge. For example, in Fig. 12–3(c), the yellow edge corresponds to a proper path marked in yellow in Fig. 12–3(b) from room "*632*" to room "*623*". According to the length and the number of turning corners of the path, a cost value is assigned to its corresponding edge in the graph. In this weighted graph, the current location of a blind user is regarded as a starting point while his/her destination is regarded as an ending point. In the navigation process, our system will use breadth first search and [**20**] to find out a path with minimum cost value. Then the navigation system will refer floor plan map and generate the corresponding ways to destination.

Both the localization and navigation processes are based on accurate scene text extraction. Fortunately, the indoor environment mostly does not contain too much background interferences, and the text information has relatively fixed pattern, e.g., room number contains only digits in print format. Thus our proposed scene text extraction can adapt its parameters to this specific environment and application. This prototype indoor navigation system is still under construction.

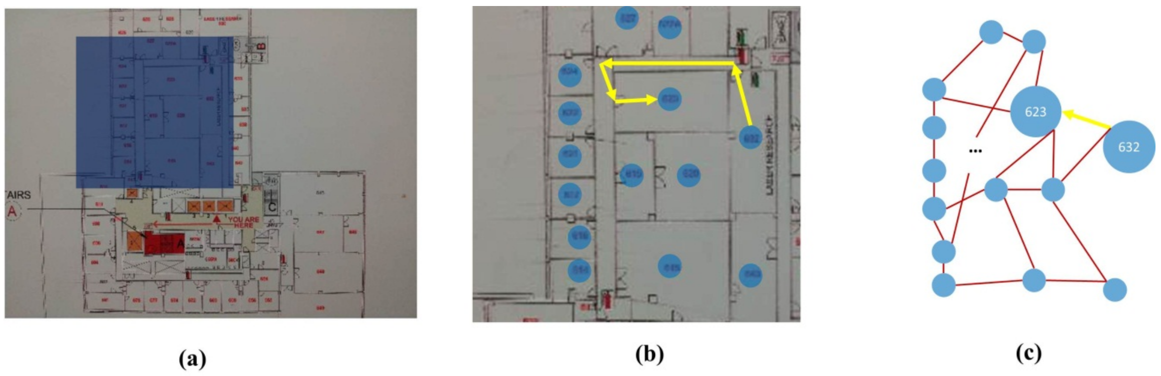


Fig. 12–3. (a) An example of floor plan, where the blue shaded region will be analyzed. (b) Each room number is regarded as a node, and a way from room "632" to room "623" is marked in yellow. (c) The abstract graph of the floor plan map, where the yellow edge indicates an edge from node "632" to node "623", corresponding the yellow way in (b).

12.3. Implementing Scene Text Extraction in Mobile Platform

To apply the scene text extraction technique in indoor navigation, we develop a prototype system in Android platform. It is able to detect regions of text strings from cluttered background, and recognize characters in the text regions in mobile device.

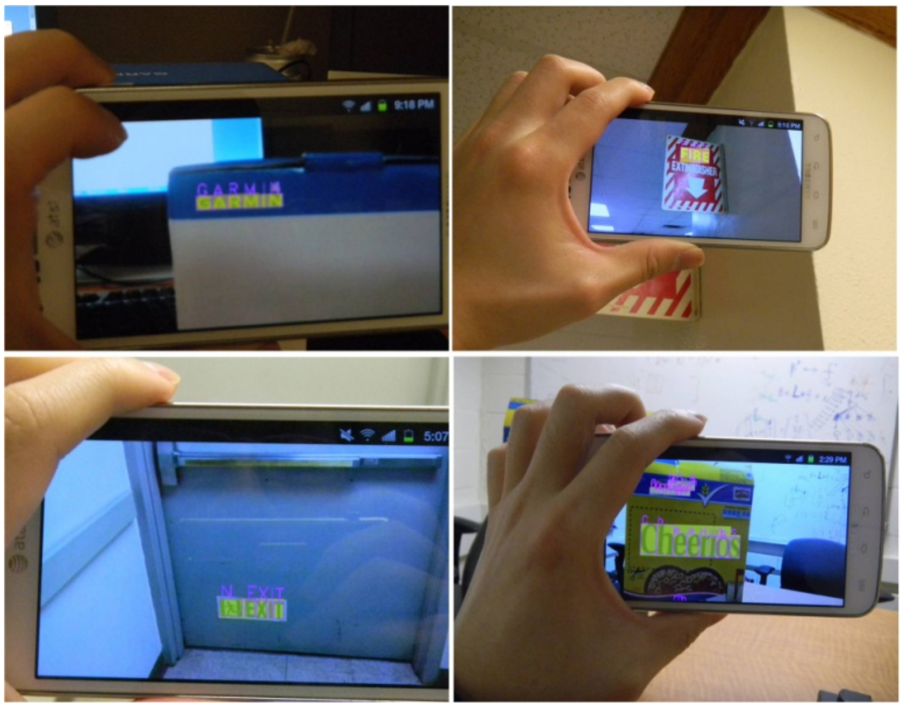


Fig. 12–4. Our prototype system of scene text extraction in Android platform. The text strings “GARMIN”, “FIRE”, “EXIT”, and “Cheerios” are extracted from complex background. Although some characters are incorrectly recognized in current prototype, we will introduce lexicon analysis to improve the recognition performance.

Compared with a PC platform as described in Section 9.2, the Android-based mobile platform is portable and more convenient to use. Scene text extraction will be more widely used in mobile applications, so it is indispensible to transplant prototype system into the popular Android mobile platform. However, two main challenges should be overcome in developing the scene text extraction application in mobile platform. First, our framework is implemented by C++ programming, while Android platform is based on Java engine. Fortunately, Android NDK is provided to compile C++ code into the Android platform. Second, due to the limitations of computing speed and memory allocation in mobile device, we attempt to make our implementations efficient enough for real applications. To improve the efficiency, we skip layout analysis of color decomposition in text detection, but directly apply the canny edge map for layout analysis of horizontal alignment. It lowers the accuracy of text detection, but is still reliable for text extraction from nearby object in enough resolutions. In addition, code optimization is performed. In our test, each frame spends about 1 second in completing the whole process of scene text extraction. One of prototype systems runs on Samsung Galaxy II smart phone with Android 2.3 as shown in Fig. 12–4. It captures natural scene by the phone camera, and extract text information online from the captured scene images, which are frame-by-frame processed.

The prototype system in Android Platform gives us some insight into algorithm design and performance improvement of scene text extraction. First, the assumptions of horizontal alignment in text layout analysis make sense in real applications. Although some text detection algorithms attempts to extract text strings in arbitrary orientations, they usually bring in more false positive text regions and lower the efficiency. However, the user can rotate the lightweight mobile devices to adaptively fit the non-horizontal text strings. Second, the accuracy of scene text detection could be improved by using the intersections of extracted text regions from consecutive frames captured by the camera at an identical scene.

12.4. SUMMARY

In this chapter, we combine scene text extraction with door detection, signage detection and floor plan parse, which generates a blind assistant system of indoor way-finding. To adopt scene text extraction in this kind of navigation system, we develop a prototype system in Android-based mobile platform.

Chapter 13 CONCLUSIONS AND DISCUSSIONS

We have discussed a complete framework of scene text extraction and its related applications by combining with other techniques. In this section, let us summarize how we address the challenges existing in scene text extraction process.

Candidate character component: To filter out most background outliers and preserve only text characters, gradient-based and color-based partition of natural scene image is performed according to the characteristic gradient distribution and color uniformity of scene text. We propose gradient-based connecting path method and color-based bigram color reduction method to extract candidate character component. Both methods work well on camera-based scene images.

Candidate string fragment: Scene text mostly appears in the form of text string, which consists of several text characters in similar size and linear alignment. Accordingly, we design adjacent character grouping and text line fitting to group the candidate character components in similar size and linear alignment. The non-text background outliers without siblings can be removed in this process, and the grouped candidate character components evolve into possible fragments of text strings.

Text structure modeling of true positive string fragments: To extract true positive text string fragments obtained from above step, structural analysis is performed on image patches of text string fragments, by extracting Haar-like features from several feature maps related to gradient distribution, stroke consistency and distribution, and edge density. Cascaded Adaboost learning model is adopted to train text classifier based on the Haar-like structural features to find out true-positive text string fragments.

Text structure modeling of scene text character for STC prediction: To recognize text information in detected text regions, we first recognize the extracted text character by designing feature representations of STC structure. We evaluate multiple feature descriptors, coding/pooling schemes and learning models to find out the most robust and discriminative feature representations of STC structure.

Hand-held object recognition: Scene text extraction is combined with motion-based background subtraction to develop a blind-assistant system of hand-held object recognition.

Tracking-based scene text recognition: Scene text extraction is combined with object tracking to improve the performance of text recognition and adapt text recognition to real applications.

LP detection: Scene text extraction is combined with car detection to localize the car LPs in camera-based scene image

Indoor navigation: Scene text extraction is used to parse floor plan map and report the current locations.

In future work, we will further improve the performance of scene text extraction from the perspective of both accuracy and efficiency. Firstly, we will design more robust algorithm of searching possible text characters on the basis of the newly-proposed MSER operator [**55**], and will design more robust algorithm of text character segmentation in both intensity and spatial level from detected text regions. Then, we will transform our framework into parallel computing model, to accelerate the whole process in real applications. Next, we will try to combine the proposed scene text extraction framework with other techniques, so that it can be more widely used in all kinds of real applications.

Appendix A. THE INVOLVED LEARNING MODELS IN OUR FRAMEWORK

A.1 ADAPTIVE BOOST

The learning process based on the Adaboost model [**88**] [**25**] is as follows.

1) Given the set of m fragment samples $(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$ where $x_i \in X$ denotes feature vector and $y_i \in \{-1,1\}$ denotes ground truth. Each fragment sample i is assigned a weight D_i, which is initialized to be 1/m.

2) In the t-th iteration, we select the optimized weak classifier $\mathrm{h_t}$ from the set of weak classifiers $\mathcal{H}$, such that $h_t = \mathrm{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{m} D_i y_i h(x_i)$, and calculate $\varepsilon_t = \sum_{i=1}^{m} D(i) \cdot (y_i \neq h(x_i))$ and $\alpha_t = 0.5 \ln((1 - \varepsilon_t)/\varepsilon_t$.

3) Update the weights by $D_i \coloneqq D_i \exp(-y_i h(x_i))$.

4) Start the next iteration from step (2) until all the fragment samples are correctly classified or the maximum number of iterations is reached.

5) The optimized weak classifiers are combined into a stage-Adaboost classifier as $H(x) = \sum_t \alpha_t h_t(x)$.

Given a testing sample x_τ, we can input it into the learned classifier H to calculate $H(x_\tau)$. Then the predicted label of x_τ is obtained from the sign of the value $H(x_\tau)$, that is, $y_\tau = 1 \;\; if \; H(x_\tau) \geq 0$ and $y_\tau = -1 \;\; if \; H(x_\tau) < 0$.

A.2 SUPPORT VECTOR MACHINE

A brief introduction of SVM model will be givenin this section. A binary SVM model computes two hyperplanes in high dimensional feature space of training samples for making classification. One hyperplane is used as the boundary of positive training samples, while the other is used as the boundary of negative training samples. An optimized pair of hyperplanes is able to separate positive samples and negative samples in minimized error, while they have maximized distance from each other.

The detail of SVM learning model for binary classification task is as follows. At first, a training dataset of positive samples and negative samples is given, denoted as $\{(x_i, y_i)|x_i \in \mathbb{R}^p, 1 \leq i \leq m\}$ where x_i represents a p-dimensional feature vector that combines all the features of the training sample i, and $y_i \in \{-1,1\}$ is the ground truth label to tell whether it is a positive sample or negative sample. Then the object is to construct a hyperplane as $w \cdot x - b = 0$, such that the positive samples fall in one side of the hyperplane and the negative samples fall into the other side of the hyperplane, while the distance from the hyperplane to the nearest samples is maximized. To solve the optimized w and b, two hyperplanes $w \cdot x - b = 1$ and $w \cdot x - b = -1$ are established. There is $w \cdot x - b \geq 1$ for positive samples and $w \cdot x - b \leq -1$ for negative samples if the samples are linear separable. If the distance between the two hyperplanes is maximized, the two classes of samples can be correctly classified while producing maximized distances to the hyperplane of each other.

From the geometrical computation, the distance between the two hyperplane boundaries is calculated by $2/\|w\|$. Thus the following task is to minimize the $\|w\|$. This problem is then transformed into the minimization of $\|w\|^2/2$ [**6**]. By the training of the hyperplane classifier, the testing sample x_j can be classified by checking whether $(w \cdot x_j - b) \geq 1$ or $\left(w \cdot x_j - b\right) \leq -1$. The primary form of SVM model is written in Eq. (A.2 – 1)

$$\arg\min_{(w,b)} \left\{ \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_n \right\} \tag{A.2 – 1}$$

$$\text{subject to } y_i(w \cdot x_i - b) \geq 1 - \xi_i, \qquad \xi_i \geq 0$$

where ξ_i is slack variable representing soft-margin of the hyperplane. By using Karush–Kuhn–Tucker conditions $w = \sum_{i=1}^{n} \alpha_i y_i x_i$ and $\sum_{i=1}^{n} \alpha_i y_i = 0$, the primary form can be transformed into dual form as Eq. (A.2 – 2).

$$\begin{gathered}\arg\max_{(\alpha)} \left\{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \right\} \\ \text{subject to } 0 \leq \alpha_i \leq C \end{gathered} \qquad \text{(A.2 – 2)}$$

where α_i is the Laplacian multiplier. $K(\cdot)$ denotes the kernel used to calculate the distance between two samples. In most cases, $K(x_i, x_j) = x_i^T x_j$ is linear kernel. Several non-linear kernels are designed to map the samples into the higher dimensional space to improve the classification performance.

Our framework adopts LIBLINEAR library [**85**] to implement the SVM training and testing for STC prediction. STC prediction is a task of multi-class classification. In training/testing process, *one-v.s.-the rest* scheme is used to implement multi-class classification under SVM model.

Appendix B. PUBLICATIONS DURING PHD STUDY

Journal papers

1. C. Yi and Y. Tian. Scene Text Recognition in Mobile Applications by Character Descriptor and Structure Configuration. IEEE Transactions on Image Processing, 2014.
2. C. Yi, Y. Tian and A. Arditi. Portable Camera-based Assistive Text and Product Label Reading from Hand-held Objects for Blind Persons. IEEE/ASME Transactions on Mechatronics, No. 99, pp. 1-10, 2013.
3. C. Yi, R. Flores, R. Chincha, and Y. Tian, Finding Objects for Assisting Blind People, Network Modeling Analysis in Health Informatics and Bioinformatics, Volume 2, Issue 2, pp 71-79, 2013.
4. S. Wang, C. Yi, and Y. Tian, "Signage Detection and Recognition for Blind Persons to Access Unfamiliar Environments," Journal of Computer Vision and Image Processing, Vol. 2, No. 2, 2012.
5. C. Yi, and Y. Tian. Text Extraction from Scene Images by Character Appearance and Structure Modeling. Computer Vision and Image Understanding, 2012.
6. C. Yi, and Y. Tian. Localizing Text in Scene Images by Boundary Clustering, Stroke Segmentation, and String Fragment Classification. IEEE Transactions on Image Processing, Vol. 21, Issue 9, 2012.
7. Y. Tian, X. Yang, C. Yi, and A. Arditi. Toward a Computer Vision-based Wayfinding Aid for Blind Persons to Access Unfamiliar Indoor Environments. Machine Vision and Applications, 2012.
8. C. Yi, and Y. Tian. Text String Detection from Natural Scenes by Structure-based Partition and Grouping. IEEE Transactions on Image Processing, Vol. 20, Issue 9, pp. 2594-2605, 2011.

Conference papers

9. X. Rong, C. Yi, X. Yang and Y. Tian. Scene Text Recognition in Multiple Frames based on Text Tracking. International Conference on Multimedia and Expo, 2014.
10. S. Joseph, X. Zhang, I. Dryanovski, J. Xiao, C. Yi, and Y. Tian. Semantic indoor navigation with a blind-user oriented augmented reality. International Conference on Systems, Man, and Cybernetics, part SMC: Human-Machine, 2013.
11. H. Pan, C. Yi and Y. Tian, A Primary Travelling Assistant System of Bus Detection and Recognition for Visually Impaired People, IEEE Workshop on Multimodal and

Alternative Perception for Visually Impaired People (MAP4VIP), in conjunction with ICME 2013.

12. C. Yi, X. Yang and Y. Tian. Feature Representations for Scene Text Character Recognition: A Comparative Study. International Conference on Document Analysis and Recognition, 2013.
13. Z. Ye, C. Yi and Y. Tian, Reading Labels of Cylinder Objects for Blind Persons, IEEE International Conference on Multimedia & Expo (ICME), 2013
14. X. Yang, C. Yi, L. Cao, and Y. Tian. MediaCCNY at TRECVID 2012: Surveillance Event Detection. In NIST Trecvid Workshop, 2012.
15. C. Yi, Y. Tian, and S. Yi, CinC Challenge: Predicting In hospital Mortality of Intensive Care Unit by Analyzing Histogram of Medical Variables under Cascaded Adaboost Model. The 39th annual meeting of computing in cardiology, 2012.
16. C. Yi, and Y. Tian. Text Detection in Natural Scene Images by Stroke Gabor Words. Proceedings of International conferences on document analysis and recognition (ICDAR), pp. 177-181, 2011.
17. C. Yi, and Y. Tian. Assistive Text Reading from Complex Background for Blind Persons. ICDAR Workshop on Camera-based Document Analysis and Recognition (CBDAR), Springer LNCS-7139, pp.15-28, 2011.
18. X. Yang, Y. Tian, C. Yi, and A. Arditi. Context-based indoor object detection as an aid to blind persons accessing unfamiliar environments. ACM International Conference on Multimedia (ACM-MM), 2010.
19. Y. Tian, C. Yi, and A. Arditi. Improving Computer Vision-Based Indoor Wayfinding for Blind Persons with Context Information. Int. Conf. on Computer Helping People with Special Needs (ICCHP), pp.255-262, 2010.

REFERENCE

[1] Abbyy. Abbyy. [Online]. http://finereader.abbyy.com/

[2] A. Akoum, B. Daya, and P. Chauvet, "A new algorithmic approach for detection and identification of vehicle plate numbers," *Journal of Software Engineering and Applications*, pp. pp. 99-108, 2010.

[3] J. Banerjee, M. Namboodiri, and C. Jawahar, "Contextual restoration of severely degraded document images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 517-524.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. Gool, "SURF: Speeded up robust features," *Computer Vision and Image Understanding*, 2008.

[5] T. Breuel, "The OCRopus open source OCR system," in *Proceedings of IS&T/SPIE Annual Symposium*, 2008.

[6] Christopher J. C. Burges, "A tutorial on support vector machine for pattern recognition," *Data Mining and Knowledge Discovery*, pp. 121-167, 1998.

[7] T. Burns and J. Corso, "Robust unsupervised segmentation of degraded document images with topic models," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1287-1294.

[8] J. Cai, H. Ji, C. Liu, and Z. Shen, "Blind motion deblrring using multiple images," *Journal of Computational Physics*, vol. 228, no. 14, pp. 5057-5071, 2009.

[9] M. Calonder et al., "BREIF: Computing a local Binary Desriptor Very Fast," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[10] caltech. (1999) caltech. [Online]. http://www.vision.caltech.edu/Image_Datasets/cars_markus/

[11] J. Canny, "A computational approach to edge detection," in *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, 1986, pp. 679-698.

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

[13] X. Chen, J. Yang, J. Zhang, and A. Waibel, "Automatic detection and recognition of signs from natural scenes," *IEEE Transactions on Image Processing*, vol. 13, no. 1, pp. 87-99, 2004.

[14] J. Chen, L. Yuan, C. Tang, and L. Quan, "Robust dual motion deblurring," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1-8.

[15] X. Chen and A. Yuille, "Detecting and reading text in natural scenes," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 13, 2004, pp. 366-373.

[16] A. Coates et al., "Text detection and character recognition in scene images with unsupervised feature learning," in *International Conference on Document Analysis and Recognition*, 2011, pp. 440-445.

[17] D. Dakopoulos and N. Bourbakis, "Wearable obstacle avoidance electronic travel aids for blind: a survey," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 40, no. 1, pp. 25-35, 2010.

[18] N. Dalal and B. Triggs, "Histogram of Oriented Gradients for Human Detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[19] T. de-Campos, B. Babu, and M. Varma, "Character recognition in natural images," in *International Conference on Computer Vision Theory and Applications*, 2009.

[20] E. Dijkstra, "A note on two problems in connexion with graph,"

Numerische Mathematik, pp. 269-271, 1959.

[21] V. Dinh, S. Chun, S. Cha, H. Ryu, and S. Sull, "An efficient method for text detection in video based on stroke width similarity," in *Asian Conference on Computer Vision*, 2007, pp. 200-209.

[22] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in Natural scene with stroke width transform," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[23] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627-1645, 2010.

[24] R. Fergus, B. Singh, A. Hertzmann, S. Roweis, and W. Freeman, "Removing camera shake from a single photograph," in *ACM SIGGRAPH*, 2006, pp. 787-794.

[25] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119-139, 1997.

[26] J. Gao and J. Yang, "An adaptive algorithm for text detection from natural scenes," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2001, pp. 84-89.

[27] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *European Conference on Computer Vision*, 2008.

[28] S. Hanif and L. Prevost, "Text detection and localization in complex scene images using constrained Adaboost algorithm," in *International Confernece on Document Analysis and Recognition*, 2009, pp. 1-5.

[29] C. Harris, "A combined corner and edge detector," in *Proceedings of the Fourth alvey vision conference*, 1988, pp. 147-151.

[30] Y. Hasan and L. Karam, "Morphological text extraction from images," *IEEE Transactions on Image Processing*, vol. 9, no. 11, 2000.

[31] ICDAR. (2003) http://algoval.essex.ac.uk/icdar/Datasets.html.

[32] ICDAR. (2011) http://robustreading.opendfki.de/wiki/SceneText.

[33] ICDAR. (2011) http://www.cvc.uab.es/icdar2011competition/.

[34] A. Jain and S. Bhattacharjee, "Text segmentation using Gabor filters for automatic document processing," *Machine vision applications*, vol. 5, pp. 169-184, 1992.

[35] S. Joseph et al., "Semantic indoor navigation with a blind-user oriented augmented reality," in *IEEE International Conference on System, Man, and Cybernetics*, 2013.

[36] K. Jung, K. Kim, and A. Jain, "Text information extraction in images and videos: a survey," *Pattern Recognition*, vol. 5, pp. 977-997, 2004.

[37] C. Jung, Q. Liu, and J. Kim, "Accurate text localization in images based on SVM output scores," *Image and vision computing*, vol. 27, no. 9, pp. 1295-1301, 2008.

[38] T. Kadir and M. Brady, "Saliency, scale, and image description," *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83-105, 2001.

[39] T. Kasar, J. Kumar, and A. Ramakrishnan, "Font and background color independent text binarization," in *Camera-based Documentation Analysis and Recognition*, 2007, pp. pp. 3-9.

[40] P. Kim, "Automatic text location in complex color images using local color quantization," in *IEEE TENCON*, 1999, pp. 629-632.

[41] KNFB Reading Technology. (2013) KReader. [Online]. http://www.knfbreading.com/

[42] S. Kumar, R. Gupta, N. Khanna, S. Chaudhury, and S. Joshi, "Text extraction and document image segmentation using matched wavelets

and MRF model," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2117-2128, 2007.

[43] S. Lefevre and N. Vincent, "Caption localization in video sequences by fusion of multiple detectors," in *International Conference on Document Analysis and Recognition*, 2005, pp. 106-110.

[44] A. Levin, Y. Weiss, F. Durand, and W. Freeman, "Efficient marginal likelihood optimization in blind devconvoltion," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 2657-2664.

[45] J. Liang, D. DeMenthon, and D. Doermann, "Geometric Rectification of Camera-captred document images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 5911-605, 2008.

[46] L. Li and M. Leung, "Integrating intensity and texture differences for robust change detection," *IEEE Transactions on Image Processing*, vol. 11, no. 2, pp. 105-112, 2002.

[47] Q. Liu, C. Jung, and Y. Moon, "Text segmentation based on Stroke Filter," in *Proceedings of International Conference on Multimedia*, 2006, pp. 129-132.

[48] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *International Conference on Computer Vision*, 2011, pp. 2486-2493.

[49] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[50] S. Lucas, "ICDAR 2005 text locating competition results," in *Interational Conference on Document Analysis and Recognition*, 2005, pp. 80-84.

[51] S. Lucas et al., "ICDAR 2003 Robust Reading Competition," in *International Conference on Document Analysis and Recognition*, 2003.

[52] S. Lu and C. Tan, "Retrieval of machine-printed Latin documents through word shape coding," *Pattern Recognition*, vol. 41, no. 5, pp. 1799-1809,

2008.

[53] C. Mancas-Thillou and B. Gosselin, "Spatial and color spaces combination for natural scene text extraction," in *IEEE Conference on Image Processing*, 2006, pp. 985-988.

[54] R. Manduchi and J. Coughlan, "Computer Vision without sigh," *Communications of the ACM*, pp. 96-104, 2012.

[55] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *British Machine Vision Conference*, 2002, pp. 384-396.

[56] L. Ma, C. Wang, and B. Xiao, "Text detection in natural images based on multi-scale edge detection and classification," in *International Congress on Image and Signal Processing*, 2010, pp. 1961-1965.

[57] A. Mishra, K. Alahari, and C. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[58] National Health Interview Survey. (2008) National Health Interview Survey. [Online]. http://www.cdc.gov/nchs/nhis/nhis_ad.htm

[59] L. Neumann and J. Matas, "A method for text localization and detection," in *Asian Conference on Computer Vision*, 2010.

[60] L. Neumann and J. Matas, "Real-time Scene Text Localization and Recognition," in *Proceedings of Computer Vision and Pattern Recognition*, 2012, pp. 3538-3545.

[61] N. Nikolaou and N. Papamarkos, "Color reduction for complex document images," *International Journal of Imaging Systems and Technology*, vol. 19, pp. 14-26, 2009.

[62] Nuance. Nuance Omnipage. [Online]. http://www.nuance.com/for-business/by-product/omnipage/index.htm

[63] K. Okuma and A. Taleghani, "A boosted particle filter multi-target detection and tracking," in *European Conference on Computer Vision*, 2004.

[64] OpenCV. OpenCV. [Online]. http://opencv.org/

[65] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on System, Man, Cyber*, pp. 62-66, 1979.

[66] W. Pan, C. Suen, and T. Bui, "Script identification using steerable Gabor filters," in *International Conference on Document Analysis and Recognition*, 2005, pp. 883-887.

[67] C. Park, K. Moon, W. Oh, and H. Choi, "An efficient extraction of character string positions using morphological operator," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2000.

[68] F. Perronnin, J. Sanchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classificaiton," in *European Conference on Computer Vision*, 2010.

[69] T. Phan, P. Shivakumara, and C. Tan, "A laplacian method for video text detection," in *International Conference on Document Analysis and Recognition*, 2009, pp. 66-70.

[70] E. Rublee, V. Rahbaoud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *International Conference on Computer Vision*, 2011, pp. 2564-2571.

[71] ScanTalker. (2006) ScanTalker. [Online]. http://www.freedomscientific.com/fs_news/PressRoom/en/2006/ScanTalker2-Announcement_3-30-2006.asp

[72] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 Robust Reading Competition," in *International Conference on Document Analysis and Recognition*, 2011, pp. 1491-1496.

[73] H. Shen and J. Coughlan, "Grouping Using Factor Graphs: an Approach for finding text with a camera phone," in *Workshop on Graph-based representation in Pattern Recognition*, 2007, pp. 394-403.

[74] M. Shi, Y. Fujisawab, T. Wakabayashia, and F. Kimura, "Handwritten numeral recognition using gradient and curvature of gray scale image," *Pattern Recognition*, vol. 35, no. 10, pp. 2051-2059, 2002.

[75] C. Shi et al., "Scene text recognition using part-based tree-structured character detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2961-2968.

[76] R. Smith, "An overview of the Tesseract OCR engine," in *International Conference on Document Analysis and Recognition*, 2007.

[77] D. Smith, J. Feild, and E. Learned-Miller, "Enforcing similarity constraints with integer programming," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[78] K. Sobottka, H. Kronenberg, T. Perroud, and H. Bunke, "Text extraction from colored book and journal covers," in *International Conference on Document Analysis and Recognition*, 1999.

[79] Y. Song et al., "A novel image text extraction method based on K-means clustering," in *International Conference on Computer and Information Science*, 2008, pp. 185-190.

[80] H. Suen and J. Wang, "Segmentation of uniform colored text from color graphics background," in *Proceedings of Vision, Image and Signal Processing*, 1997, pp. 317-332.

[81] Technologies for the Visually Impaired Inc. (2013) Technologies for the Visually Impaired Inc. [Online]. http://www.tvi-web.com/

[82] L. Tian, C. Yi, and Y. Tian, "Detecting good quality frames in videos captured by a wearable camera for blind navigation," in *IEEE Conference*

on Bioinformatics and Biomedicine, 2013, pp. 334-337.

[83] E. Tola, A. Fossati, C. Strecha, and P. Fua, "Large occlusion completion using normal maps," , 2010.

[84] H. Tran, A. Lux, L. Nguyen, and A. Boucher, "A novel appproach for text detection in images using structural features," in *International Conference on Advances in Pattern Recognition*, 2005, pp. 627-635.

[85] Machine Learning Group at National Taiwan University. LIBLINEAR -- A Library for Large Linear Classification. [Online]. http://www.csie.ntu.edu.tw/~cjlin/liblinear/

[86] A. Vedaldi. VLFeat.org. [Online]. http://www.vlfeat.org/

[87] A. Vedaldi and A. Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 480-492, 2012.

[88] P. Viola and M. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, 2004.

[89] K Wang. (2010) http://vision.ucsd.edu/~kai/svt/.

[90] K. Wang, B. Bbenko, and S. Belongie, "End-to-end scene text recognition," in *International Conference on Computer Vision*, 2011, pp. 1457-1464.

[91] K. Wang and S. Belongie, "Word spotting in the wild," in *European Conference on Computer Vision*, 2010, pp. 591-604.

[92] J. Weinman, A. Hanson, and A. McCallum, "Sign detection in natural images with conditional random fields," in *IEEE International Workshop on Machine Learning for Signal Processing*, 2004.

[93] J. Weinman, E. Learned-Miller, and A. Hanson, "Scene text recognition using similarity and a lexicon with sparse belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1733-1746, 2009.

[94] WHO. (2009) 10 facts about blindnes and visual impairment. World Health Organization: Blindness and visual impairment.

[95] C. Wolf, J. Jolion, and F. Chassaing, "Text localization, enhancement and binarization in multimedia documents," in *International Conference on Pattern Recognition*, 2002, pp. 1037-1040.

[96] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors," *International Journal of Computer Vision*, vol. 75, no. 2, pp. 247-266, 2007.

[97] X. Yang and Y. Tian, "Robust Door Detection in Unfamiliar Environments by Combining Edge and Corner Features," in *IEEE Conference on Computer Vision and Pattern Recognition Workshop on Computer Vision Applications for Visual Impaired*, 2010.

[98] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1794-1801.

[99] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting text of arbitrary orientations in natural images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[100] Z. Ye, C. Yi, and Y. Tian, "Reading labels of cylinder objects for blind persons," in *International Conference on Multimedia and Expo*, 2013, pp. 1-6.

[101] X. Yin, K. Huang, and H. Hao, "Robust text detection in natural scene images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[102] C. Yi and Y. Tian, "Text detection in natural scene images by stroke Gabor words," in *International conference on document analysis and recognition*,

2011, pp. 177-181.

[103] F. Zhang, H. Cheng, Q. Liu, and M. Wan, "Text localization in span image using edge features," in *International conference on Communications, Circuits and Systems*, 2008, pp. 838-842.

[104] J. Zhang and R. Kasturi, "Extraction of text objects in video documents: recent progress," in *IAPR International Workshop on Document Analysis Ssytems*, 2008, pp. 5-17.

[105] Q. Zheng, K. Chen, Y. Zhou, G. Cong, and H. Guan, "Text localization and recognition in complex scenes using local features," in *Asian Conference on Computer Vision*, 2010, pp. 121-132.