12-9-2013 12:00 AM

Detecting Multilingual Lines of Text with Fusion Moves

Igor Milevskiy, *The University of Western Ontario*

Supervisor: Professor Yuri Boykov, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

DETECTING MULTILINGUAL LINES OF TEXT WITH FUSION MOVES

(Thesis format: Monograph)

by

Igor Milevskiy

Graduate Program in Computer Science

A thesis submitted in partial fulfillment

of the requirements for the degree of

Masters of Science

The School of Graduate and Postdoctoral Studies

The University of Western Ontario

London, Ontario, Canada

Abstract

This thesis proposes an optimization-based algorithm for detecting lines of text in images taken by hand-held cameras. The majority of existing methods for this problem assume alphabet-based texts (e.g. in Latin or Greek) and they use heuristics specific to such texts: proximity between letters within one line, larger distance between separate lines, etc. We are interested in a more challenging problem where images combine alphabet and logographic characters from multiple languages where typographic rules vary a lot (e.g. English, Korean, and Chinese). Significantly higher complexity of fitting multiple lines of text in different languages calls for an energy-based formulation combining a data fidelity term and a regularization prior. Our data cost combines geometric errors and likelihoods given by a classifier trained to low-level features in each language. Our regularization term encourages sparsity based on *label costs*. Our energy can be efficiently minimized by *fusion moves*. The algorithm was evaluated on a database of images from the subway of metropolitan area of Seoul and was shown to be robust.

Keywords: Object Detection, Applications of Computer Vision, Discrete optimization

Acknowledgments

I would like to express my appreciation to my advisor, Dr. Yuri Boykov. for his guidance, encouragement and enthusiasm.

I appreciate the advice of my thesis committee, professors John Barron, Charles Ling and Serguei Primak.

Thanks to Dr. Olga Veksler for her comments about text classification. Thanks to Andrew Delong, Lena Gorelick and Hossam Isack for sharing their knowledge in graph theory and optimization. I would like to thank Dr. Jin-Young Ha for opening the field of pattern recognition and computer vision to me. Thanks to Janice Wiersma and Cheryl McGrath for their help and support.

Finally, thanks to my brother, Vladislav, who invited my for the 2009-10 New Year celebration in Tokyo and left me behind somewhere between Shinjuku, Shibuya and Tsurukawa. One day in 2010 played a tremendous role in this thesis motivation. I would like to thank my parents in Russia, who has always given me their support, despite me being far away for the last four years. My time at the University of Western Ontario has been exiting. I have learned and experienced lots of things not just in the major but also in the Canadian culture.

Contents

List of Figures

List of Algorithms

List of Tables

List of Appendices

Chapter 1

Introduction

Over the centuries, people in different parts of the world developed various writing systems. The most common writing systems are Latin, Cyrillic, Arabic, Logographic Chineese characters, such as Hanzi, Haja and Kanji (in use in China, Korea, and Japan respectively), and alphabet based Hangul (Korea), Kitakana and Hiragana (both Japan). Most people comprehend text in one or at most two writing systems. Thus, it is extremely helpful to dedicate reading functions to a computing device. Significant progress in text detection and recognition was made in the scope of printed documents. Recently, researchers focus on more challenging tasks, such as text detection in video, web, and camera captured images [8, 20, 22, 39], which is also the main focus of this work.

For a single line of text (e.g. in on-line handwriting recognition) or when multiple lines are easy to detect (e.g. in printed documents) the problem of text recognition can be solved with well established techniques such as the Hidden Markov Model (HMM) and Recurrent Neural Networks (RNN) [10]. The input for these algorithms is an image segment corresponding to one line of text. In particular, this segment does not need to be split into isolated characters.

In the context of camera captured images, e.g. Fig.1.1, detection of text lines is an additional challenging problem that must be solved before image segments containing separate lines can be sent to the recognizer. Accurate partitioning of multiple text lines is critical for

Figure 1.1: Signboard in three languages. Korean and English text lines are marked with boxes. Two Korean character form one word. There is a significant gap between them, which makes the text detection problem challenging.

the success of the recognizer. This paper focuses specifically on the task of individual text line recognition, which is particularly challenging in presence of multiple languages, see Fig.1.2(c).

Most of the prior work on text line detection deals with scripts in Latin characters printed according to one standard set of typographical rules. In Asia, however, signboards on streets, in subways stations, railway platforms and bus terminals can have text in two or more languages at once. We aim to detect multiple lines of text from an image, e.g. Fig.1.1, that has characters in several languages arranged according to different typographical regulations.

For example, English typography includes letters in upper and lower cases, some characters also have descenders and ascenders. English letters can cross the median and the baseline. One common convention is that English letters sit close to each other within each line of text, which makes text line detection relatively easy.

In contrast, Korean typographical rules make it harder to detect lines of text for two reasons. The first one is illustrated in Fig.1.1 where two green boxes mark Korean characters (or syllabuses) belonging to the same word. The gap between these characters is significantly

greater than the distance between the first Korean character and the English line below it. Such large inter-character gaps are very common in Korean typography.

The second difficulty is that multiple Korean letters can sit on top of each other within a single text line. For example, each green box in Fig.1.1 corresponds to a syllabus containing 3 letters. As shown in Fig.1.2(b), it is easy to incorrectly group individual Korean letters in the same syllabus into multiple lines. Fig.1.2(c) shows the correct segmentation.

Similarly to Korean, Chinese characters often disperse into pieces. Thus, it could be hard to determine if one piece (a detected blob) corresponds to a character or a part of a character. Another similarity between Chinese and Korean characters distinguishing them from English letters is their fairly consistent aspect ratio, typically around one.

Multilingual text detection in camera-captured images represents a particular case of scene understanding over a small set of object classes (languages). The general scene understanding problem (e.g. on PASCAL data) is commonly solved by partitioning an image into segments of different classes, e.g.[14]. However, most methods do not separate distinct object instances within one class[1]. The technical challenge of our multilingual text detection problem is that we have to segment an image into individual instances of objects (text lines) from multiple categories (languages). We formulate this as a multi-model fitting problem based on *label cost* regularization [6]. The original image is represented by a set of detected blobs, which are assigned different models/labels. Each model (text line) is described by geometric parameters[2] and an additional category parameter - language. This category defines how the data fitting errors are computed accounting for typographical differences between the languages.

There is a lot of prior work on geometric model fitting in vision. RANSAC is the most common method for data supporting a single model. It is known for its robustness to outliers. Multi-model problems are often addressed by procedurally-defined clustering heuristics greedily maximizing inliers, e.g. Mutli-RANSAC [41], Hough space mode selection, or *j-linkeage* [33]. These methods often fail on difficult examples with weak data in the presence of much noise or

[1] Some methods use object detectors to count object instances [15]

[2] Due to perspective distortion we represent *base-* and *mean lines*.

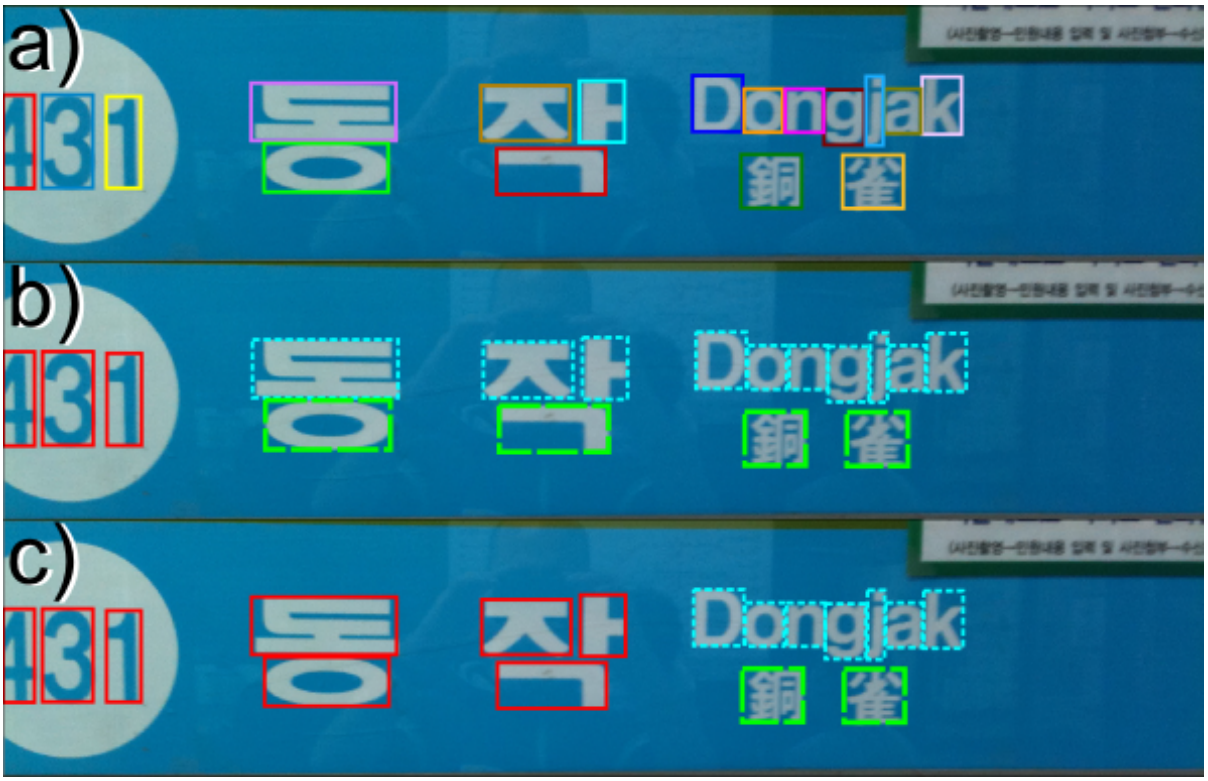


Figure 1.2: Korean characters introduce ambiguity to text line detection. a) Boxes show isolated blobs detected in the image. b) Blobs are split into text lines incorrectly: two Korean words in the center are cut. c) Correct text lines: one line on the left and two lines on the right.

outliers [12]. The practical challenges of multilingual multi-line text detection motivates more powerful approaches based on optimization of a clearly defined objective function, in particular, MDL criteria [6, 34]. Many previous MDL methods fit geometric models of the same class, e.g. lines. Some techniques fit a hierarchy of geometric models like *points→lines→cubes* [28] or *points→lines→vanishing points* [7]. We approach the multilingual text line detection problem by fitting geometric models (lines) from independent appearance-based categories (language) that can be interpreted as a hierarchy *blobs→lines→languages*.

Our contributions are summarized below:

- We propose a new challenging application for computer vision: multilingual text line detection over languages with different typography rules. Our database of camera-captured multilingual text images with ground truth will be made publicly available.

- We propose a hierarchical MDL energy for multilingual text detection (6.5). Our sparsity-based (label cost) regularization is applicable to a wider range of typographies compared to smoothness-based approaches [24, 32] assuming proximity between letters. Our energy can be efficiently optimized by fusion moves [7].

- In contrast to many standard techniques for scene understanding, our method simultaneously segments an image into multiple classes (languages) and into individual object instances (text lines) within each class.

- We propose a color based energy minimization approach for blob detection in signboards. Our method is robust in cases when camera failed to focus properly. Moreover, it works well in image contains artifacts such as reflections, glare, and shadows.

The rest of the thesis is organized as follows. Chapter 2 discusses related prior work on text detection and model fitting including energy minimization model fitting framework. Chapter 3 briefly describes our overall algorithm for text line detection. Chapter 4 contains overview of blob detection methods and our novel color-based energy minimization approach. Chapter 5 describes the classifier and features used in our current work. Chapter 6 presents our new hierarchical MDL formulation of the multilingual generalization of this problem. We also overview standard fusion moves [7] applicable to efficient optimization of such energies. Our camera-captured text image database and experimental evaluation of our algorithm is presented in Chapter 7.

Chapter 2

Related work

2.1 Text detection

Interest in text detection has increased since 2003 when the ICDAR database and competition was formed. After that, ICDAR 2011, Street View Text(SVT) and private publicly unavailable databases were collected. All mentioned databases include text in Latin script. A number of text detection algorithms were proposed and evaluated on these image collections.

In general, a text detection algorithm combines the following parts: text candidate detection, text candidate filtering and line fitting. For a given input image the algorithm produces a set of rectangles either horizon-oriented or rotated.

There are three major groups of text candidate detection methods: sliding window [4, 8, 13, 20, 24, 25, 40], edge based [26, 27, 32], and color based[21, 22] algorithms.

Sliding window algorithms, originally proposed for face detection, denote an exhaustive search. Features are computed for each position and scale of the sliding window.

Edge based methods retrieve an edge map (Sobel, Canny, Laplace) and then perform connected component (CC) analysis and outputs blobs. Moreover, Stroke width transform (SWT) [8, 39]also aims to find blobs that have consistent width of stroke. Color based methods, such as MSER and ER (inspired by MSER) assume that a text character's color is homogeneous.

MSER was used by the "ICDAR 2011 robust reading competition" winner.

After text candidates are detected non-text blobs must be filtered out. The decision whether a blob represents text is done by classification. Popular classifiers are: support vector machine (SVM) [19, 21, 26] , AdaBoost [16, 30] or their cascades. Popular features for classification are: color based (histogram of intensities, moment of intensity) edge based (histogram of oriented gradients (HOG), Gabor filter) geometrical features (width, height, aspect ration, number of holes, convex hull, area of background/foreground).

Single text blobs must then be aggregated into text strings. Older approaches are based on the Hough transform algorithm [3, 31]. More recent algorithms combine neighbours of blobs into pairs and then fulfill clustering in N-dimensional space, where the following dimensions are in use: stroke width, orientation of a pair, and geometrical size of blobs.

Text candidate filtering and line fitting could be done simultaneously with a complex approach based on a Markov random field (MRF) [20, 25] or a Conditional Random Field (CRF) [23]. The neighbour map could be constructed by a minimal spanning tree algorithm. The unary term reflects the likelihood of a blob to be text. The pairwise term describes how well two blobs constitute a line (or a part of a line).

In [2, 3] the authors assume that text is located on a signboard. They try to locate physical edges and corners of that signboard and perform signboard rectification.

2.2 Model fitting

Model fitting is the procedure of quantifying models, determining inliers for each model, and estimating model parameters. In this thesis we use an energy-based formulation for multi-model fitting that was previously applied to many types of geometric models: lines, line segments, circles, planar homographies, fundamental matrices, etc. We use the following notation. Assume that $\mathcal{I}$ is the set of all data points i and $\mathcal{L}$ is a set of indexes enumerating all possible

models. Let vector $\boldsymbol{l}$ denote indexes of models assigned to all data points i:

$$\boldsymbol{l} = \{l_i \in \mathcal{L} | i \in \mathcal{I}\}. \tag{2.1}$$

Vector $\boldsymbol{\theta}$ defines parameters for all models.

$$\boldsymbol{\theta} = \{\theta_j \in \Theta | j \in \mathcal{L}\}, \tag{2.2}$$

where Θ is the space of parameters for each model. For instance, in case of 2D line fitting the parameter search space is $\theta_j \in [0, \pi) \times \mathbb{R}_{\geq 0}$.

The objective function for multi-model fitting problems could be formulated as in [6, 12]

$$E(\boldsymbol{l}, \boldsymbol{\theta}) = \sum_{i \in \mathcal{I}} D_i(l_i | \boldsymbol{\theta}) + \sum_{j \in \mathcal{L}} C_j \cdot [\exists l_i = j], \tag{2.3}$$

where $D_i(j|\boldsymbol{\theta})$ is the data fidelity/error term describing how well each data point i fits model j

$$D_i(j|\boldsymbol{\theta}) = \|i - \theta_j\| \tag{2.4}$$

for some error measure $\|\cdot\|$ and constant C_j is a fixed penalty for each model in the solution. Penalty C is incurred if there is at least one data point i assigned to model j. The second term in (2.3), a.k.a. *label cost*, encourages sparsity. That is, it prefers solutions with fewer models.

Consider a simple model fitting example in Fig.2.1 where the models are lines (red and green) and the data points are dots. The color of each dot i describes model assignment l_i. Each 2D line model θ could be described by three homogeneous parameters

$$\theta = (A, B, C). \tag{2.5}$$

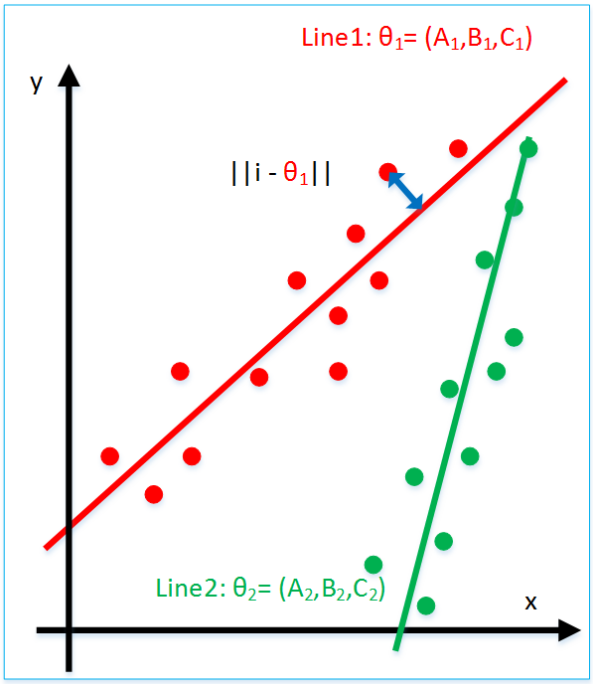


Figure 2.1: Line fitting. The dots are data points. The red and the green lines are models. The blue line shows data term (euclidean distance) from the point and the red line.)

The equation for line θ through point (x, y) is

$$Ax + By + C = 0. \tag{2.6}$$

In this case data term (2.4) could be defined by Euclidean (or orthogonal) error measure $\|\cdot\|$

$$\|i - \theta_j\| = \frac{|A_j x_i + B_j y_i + C_j|}{\sqrt{A_j^2 + B_j^2}} \tag{2.7}$$

illustrated in multi-line fitting solution in Fig.2.1.

Different multi-model fitting cost functions like Eq. (2.3) were previously addressed with a number of methods such as EM [34], quadratic pseudo-boolean optimization (QPBO) [7], linear programming(LP) relaxation [17], greedy uncapacitated field location (UFL) [6] and graph cut [6, 12].

Iscack et al. [12] used block-coordinate descent (BCD) for optimization of multi-model

fitting energy (2.3)[1]

$$\arg\min_{\boldsymbol{l},\boldsymbol{\theta}} E(\boldsymbol{l},\boldsymbol{\theta}) \tag{2.8}$$

This is NP-hard mixed optimization problem combining integer- and real-valued parameters, correspondingly $\boldsymbol{l}$ and $\boldsymbol{\theta}$. Nevertheless, in practice, block-coordinate descent for energy (2.3) often converges to good solutions that are close to the global minimum.

The following Algorithm 1 is used in [12]. First, the algorithm generates a finite set of models by randomly sampling data points. Next, BCD for energy (2.8) iterates two steps: model assignment $\boldsymbol{l}$ optimization and model parameters $\boldsymbol{\theta}$ tuning. The resulting set of models is a local minimum of objective function (2.3) with respect to block-coordinate descent (BCD) moves.

Algorithm 1: PEARL // ENERGY-BASED MODEL FITTING ALGORITHM

Input: *data_points* // pixels, edges, blobs, etc.
Output: *labeling* // data_points-to-model map
models // planes, lines, text lines

```
pool ← RandomSampleModels(data_points);
for i ← 0 to max_iterations do
    labeling ← AssignModels(pool, data_points);
    pool ← RefitModels(pool, data_points, labeling);
models ← pool
return [models, labeling];
```

We use this general approach to multi-model fitting for two novel applications: signboard fitting and text line detection.

[1] If the number of models is fixed to K instead of imposing a soft sparsity constraint, *i.e.* the label cost term in (2.3), then BCD for the data term in (2.8) is equivalent to *K-means* clustering.

Chapter 3

Overview of our approach

The algorithm work flow is shown in Fig. 3.1. First, in the input image blobs are detected. Next, AdaBoost classifiers provides the likelihood that each blob belongs to one of the five categories (English, Korean, Chinese, Digit, Non-text). Finally, text line aggregation is performed. During the last step all blobs participate in text line aggregation. Classification likelihood is used as an additional parameter, which describes how close a certain text candidate is to be assigned to a particular text line.

We use energy-based model fitting Algorithm 1 for two particular problems: blob detection (Section 6.1.1) and text line aggregation (Chapter 6). The current blob detector is described in Section 4.1.1. We plan to integrate the novel blob detection method in future work.

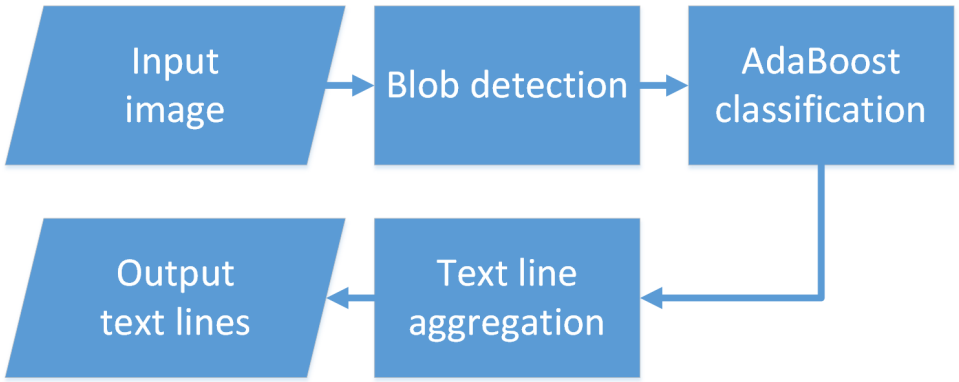


Figure 3.1: Text line detection algorithm.

Chapter 4

Blob detection

4.1 Overview and related work

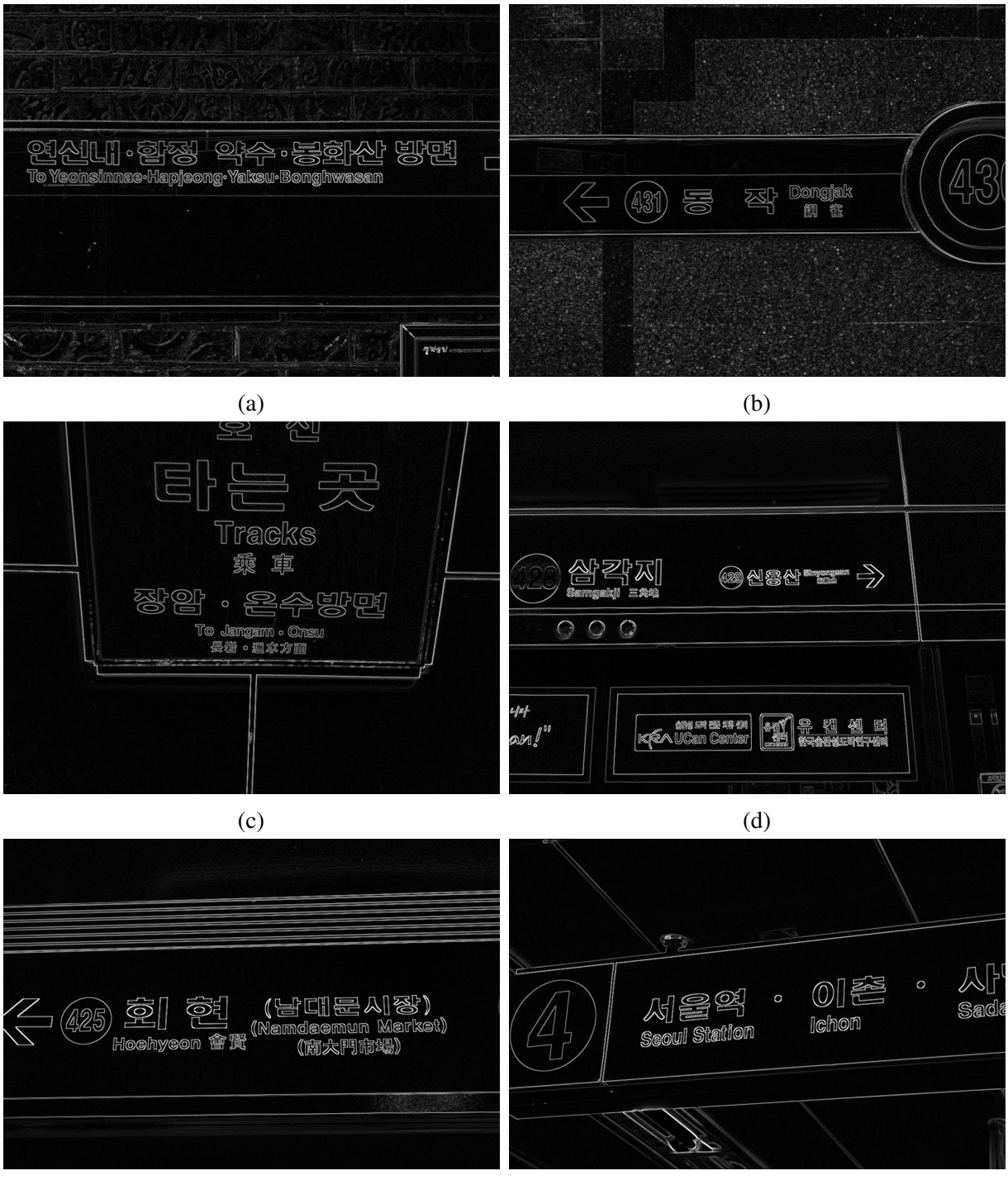
4.1.1 Edge-based approach

Signboards are designed to have high contract between text and background. Edge-based method creates edge map, thresholds the result with some fixed level, and then run connected-component analysis.

Edge maps can be obtained by applying the Laplace or Sobel filters, estimating Difference of Gaussians (DoG) or Laplacian of Gaussian (LoG). In order to detect blobs that stand close to each other and the Canny or a ridge detector can be applied.

Currently, we use an edge based blob detector to produce text candidates. At first, an input image is converted to gray-scale and downscaled for performance reasons. Next, horizontal and vertical Sobel filters are applied. Two resulting images are combined. The result represents an edge map (Fig. 4.1). Finally, connected component analysis run resulting character candidates (blobs) (Fig. 4.2).

Unlike a color based blob algorithm, an edge based method does not have foreground-background ambiguity, where the algorithm must be run twice for dark text on a bright background or vice versa.

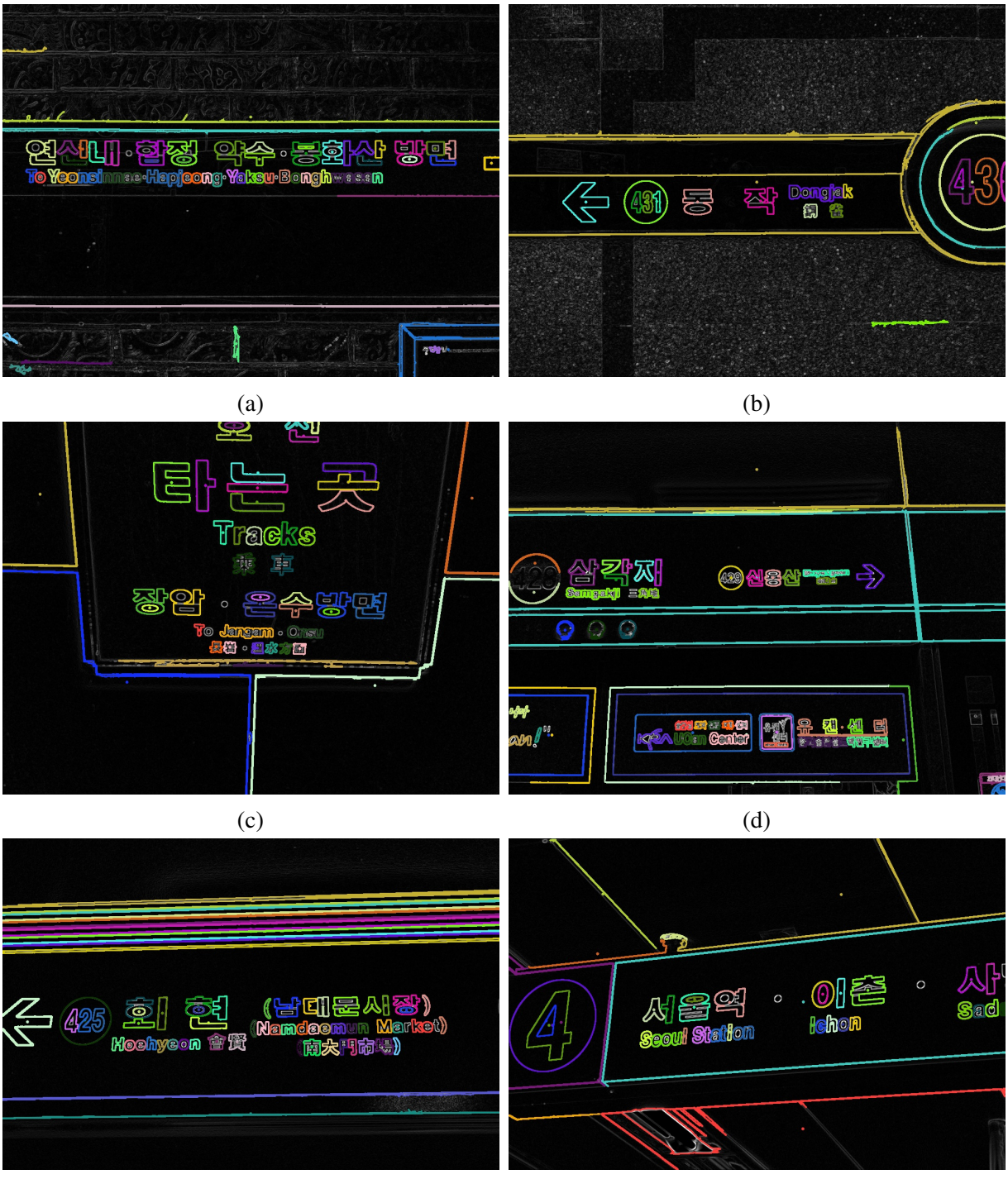


(a) (b)

(c) (d)

(e) (f)

Figure 4.1: Sobel edge maps. White pixels represent strong edge. Black pixel show weak edge.



(a) (b)

(c) (d)

(e) (f)

Figure 4.2: Blob detection from sobel maps. The result is shown as color contours. A contour in one color represents one blob.

4.1.2 Color-based approach

The color-based approach assume that the text is significantly brighter than its the background or vice versa. The color-based approach takes in an image, runs thresholding (binarization), and then performs connected component analysis.

Adaptive thresholding methods are commonly used for text binarization in printed documents [11]. The algorithms work as following. For each pixel, the mean intensity μ and the standard deviation of intensities σ, are estimated inside a window of a fixed size around that pixel. Next, the decision whether the pixel belongs to the background or text is made by binarization is performed while threshold level is calculated according to Eq. 4.1 (Niblack method) or Eq. A.5 (Sauvola method). Fig. 4.3 shows an example of adaptive thresholding.

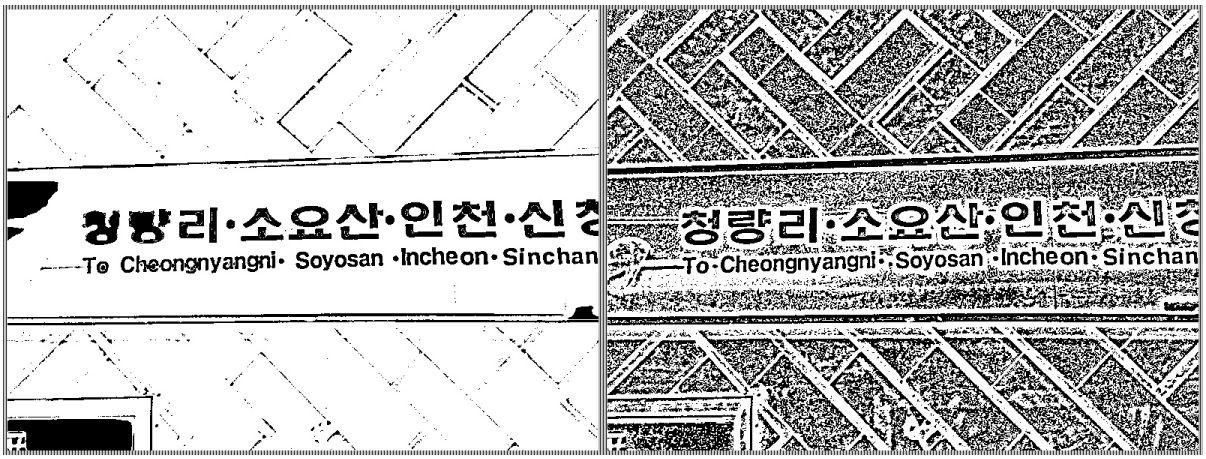
$$T = \mu + k\sigma \tag{4.1}$$

$$T = \mu\left(1 - k\left(1 - \frac{\sigma}{R}\right)\right), \tag{4.2}$$

where μ is mean, σ is standard deviation of the intensities scale level inside a window around a pixel, R is a coefficient (usually equals to 128), and k is a predefined coefficient and is selected according to text color. If the text color is darker than background than k has a negative value, otherwise it is positive.

Adaptive thresholding can be effectively implemented by integral images (see more details in Appendix A.2). In such case algorithms' complexity does not depend on the window size and is constant. Unlike an edge-based approach, adaptive binarization calculates the threshold for each pixel.

The method has to be run twice, assuming the text is brighter than the background and vice versa. It is not clear, a-priori, what coefficient k, and, what window size would give the best results.



(a) (b)

Figure 4.3: Example of adaptive thresholding. a) Niblack thresholding b) Sauvola thresholding.

4.2 Our approach. Signboard fitting

We propose a color-based energy minimization method for blob detection. The goal is to fit a signboard model onto image. Knowing models and labeling thresholding becomes simple.

We use the following notation. Assume that $\mathcal{I}$ is the set of all pixels i in the image and $\mathcal{L}$ is a set of indexes enumerating all possible signboard models. Let vector $\boldsymbol{l}$ denote indexes of signboard models assigned to all the pixels Eq (2.1). Vector $\boldsymbol{\theta}$ defines parameters for all models Eq (2.2).

4.2.1 Objective function

The objective function for signboard multi-model fitting problems could be formulated as

$$E(\boldsymbol{l}, \boldsymbol{\theta}) = -\sum_{i \in \mathcal{I}} ln(Pr(i|l_i, \boldsymbol{\theta})) + \sum_{j \in \mathcal{L}} C_j \cdot [\exists l_i = j], \tag{4.3}$$

where $Pr(i|l_i, \boldsymbol{\theta})$, is the probability of observed pixel i belonging to a signboard model l_i, C penalizes the number of signboards in the solution.

A signboard model l_i consist of three planes $\lambda = (\lambda^L, \lambda^a, \lambda^b)$ - one for each channel in the

Lab color space and three standard deviations $\sigma = (\sigma^L, \sigma^a, \sigma^b)$.

A plane is defined by four parameters $\lambda = (A, B, C, D)$:

$$Ax + By + Cz + D = 0, \tag{4.4}$$

where x, y is pixel's location and z is value of a certain channel.

The data term of energy Eq. (4.3) is:

$$\begin{aligned} &-ln(Pr(p|j, \boldsymbol{\theta})) = \\ &\frac{\|p - \lambda_j^L\|^2}{2(\sigma_j^L)^2} + ln(\sigma_j^L) + \\ &\frac{\|p - \lambda^a\|^2}{2(\sigma_j^a)^2} + ln(\sigma_j^a) + \\ &\frac{\|p - \lambda^b\|^2}{2(\sigma_j^b)^2} + ln(\sigma_j^b), \end{aligned} \tag{4.5}$$

where $\|p - \lambda\|$ is the Euclidean distance from a pixel $p = (x, y, z)$ to a plane $\lambda = (A, B, C, D)$:

$$\|p - \lambda\| = \frac{|Ax + By + Cz + D|}{\sqrt{A^2 + B^2 + C^2}}. \tag{4.6}$$

A signboard model can describe a surface with a slant. For instance, a signboard can have one side significantly brighter than the other (Fig. 4.4 a), upper signboard).

4.2.2 Optimization details

Signboard fitting Algorithm's 1 input is an image pixels (data point) in Lab color space. The Algorithm 1 outputs the set of signboard (models) and labeling. Labeling is a map that links image pixels and signboard models.

Signboard models are proposed as following. First, super pixels are found in the original image by an algorithm [9]. Next, each super pixel produces a model. All three planes in L,a and b channels are fit by linear least squares (LS) giving three sets of A, B, C, D coefficients and

σ's (See more details on LS algorithm in Appendix A.1). The model assignment is performed by Greedy-UFL algorithm [6].

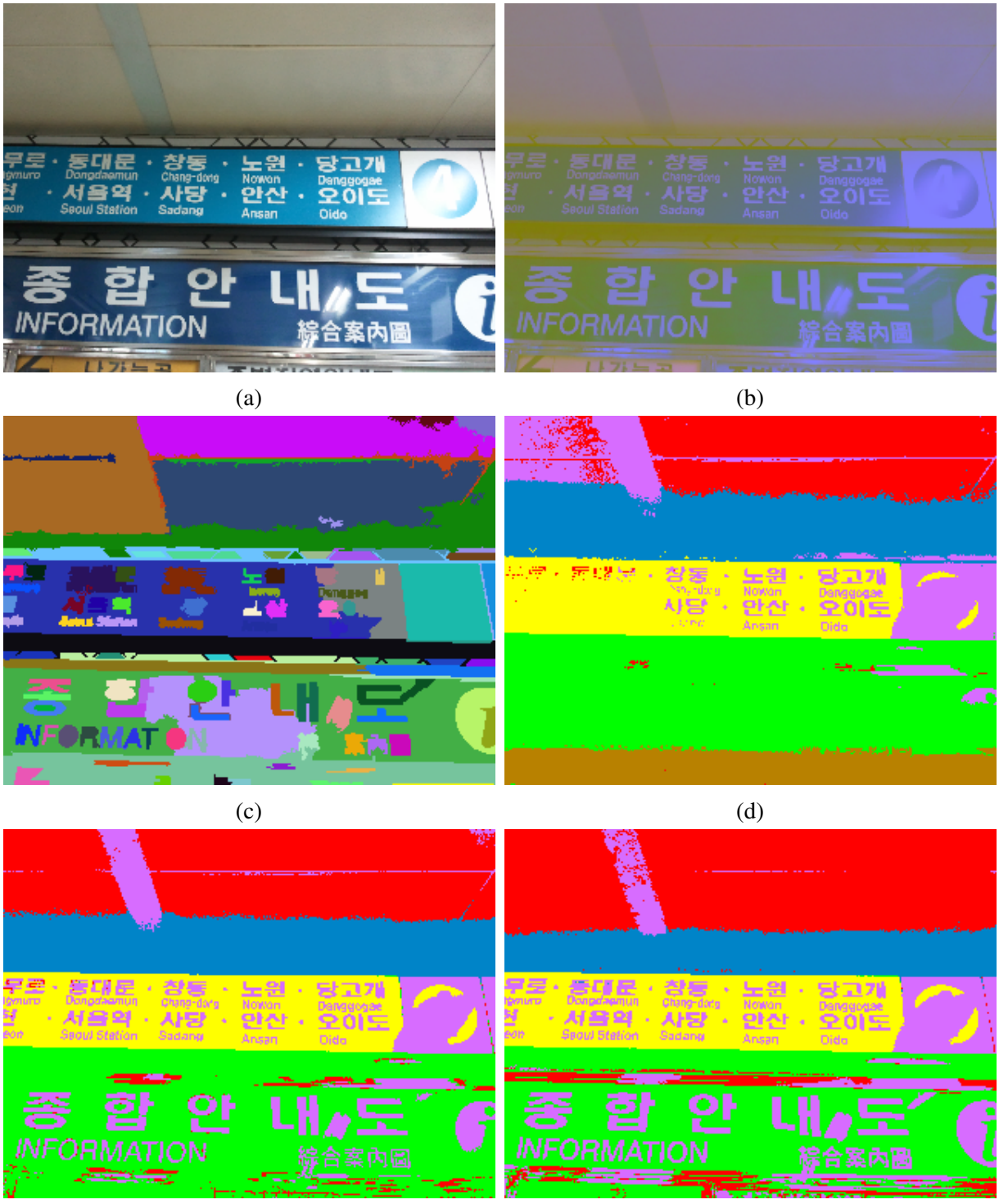
On the model refit step the labeling is fixed. One signboard is fit to the cloud of pixels assigned to that model in the same manner as during the model proposal step. After this step all model's coefficients, including σ, are updated.

The results of signboard fitting are shown in Fig. 4.4 and Fig. 4.5 - Korean subway signboard, Fig. 4.6 - signboard with reflection artifacts, Fig. 4.7 - Japanese railway station signboard, and Fig. 4.8 shopping mall signboards. In the experiments all tunable parameters are fixed.

Like adaptive thresholding's μ, a plane equation represents set of individual μ's for a group of pixels. Fig. 4.9 shows comparison of edge-based blob detection and our approach. The camera failed to focus correctly on the signboard, and so image is blurry. The edges of text are weak and were not detected (Fig. 4.9 b). Energy-based signboard fitting enables correct blob detection in this challenging example (Fig. 4.9 c).

4.2.3 Blob proposals

After text candidates are found, there are Korean and some Chinese characters that are oversegmented. It is necessary to propose more text candidates to cover such characters. An example is shown in Fig. 4.10. Two or three blobs produce a new blob if they stand close to each other, cover resulting area well and the resulting aspect ratio is close to one.

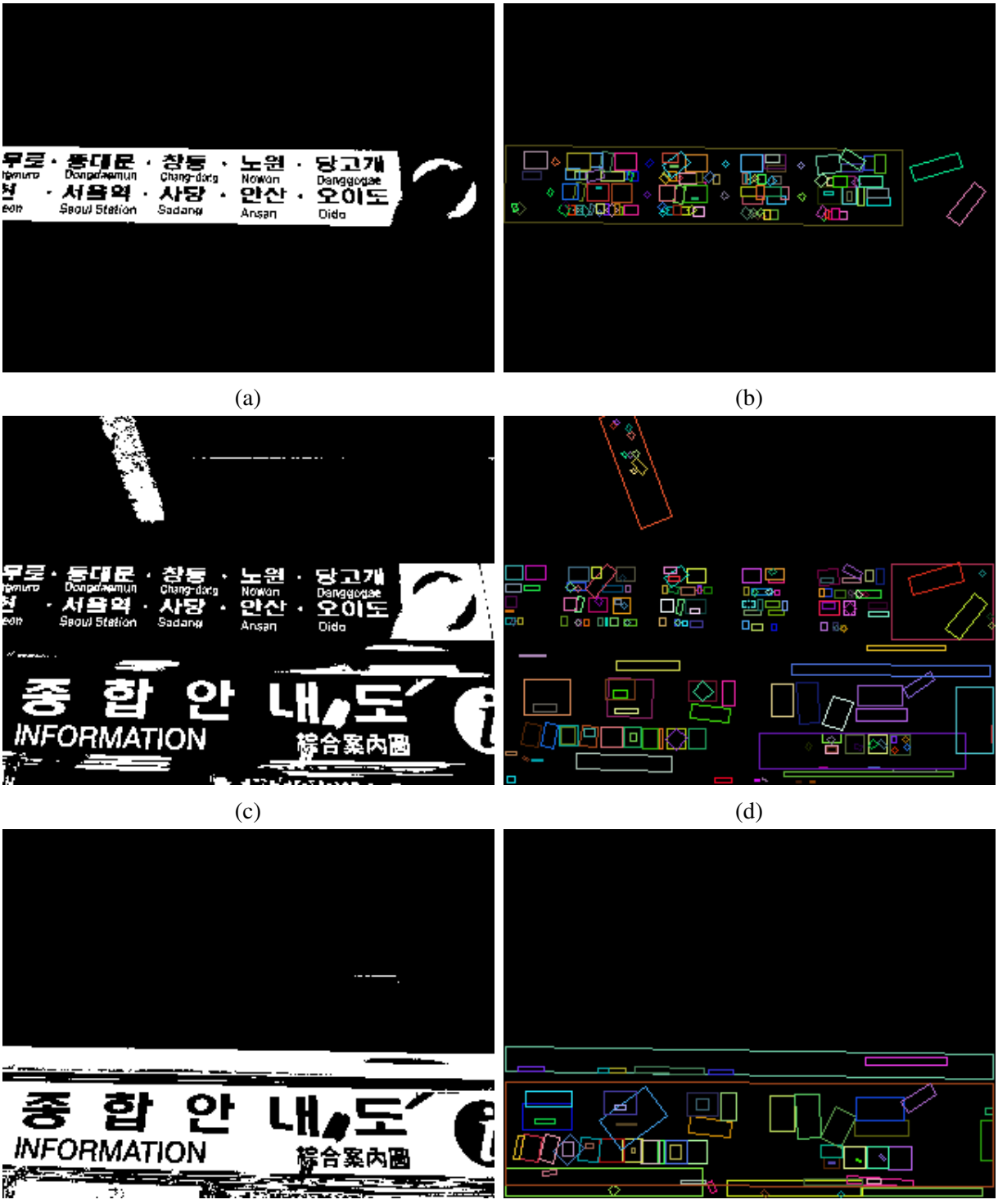


(a) (b)

(c) (d)

(e) (f)

Figure 4.4: Signboard fitting. Example 1. a) input image, b) image in Lab color space, c) super pixels, d) first iteration of signboard fitting, e) intermediate iteration and f) final result.

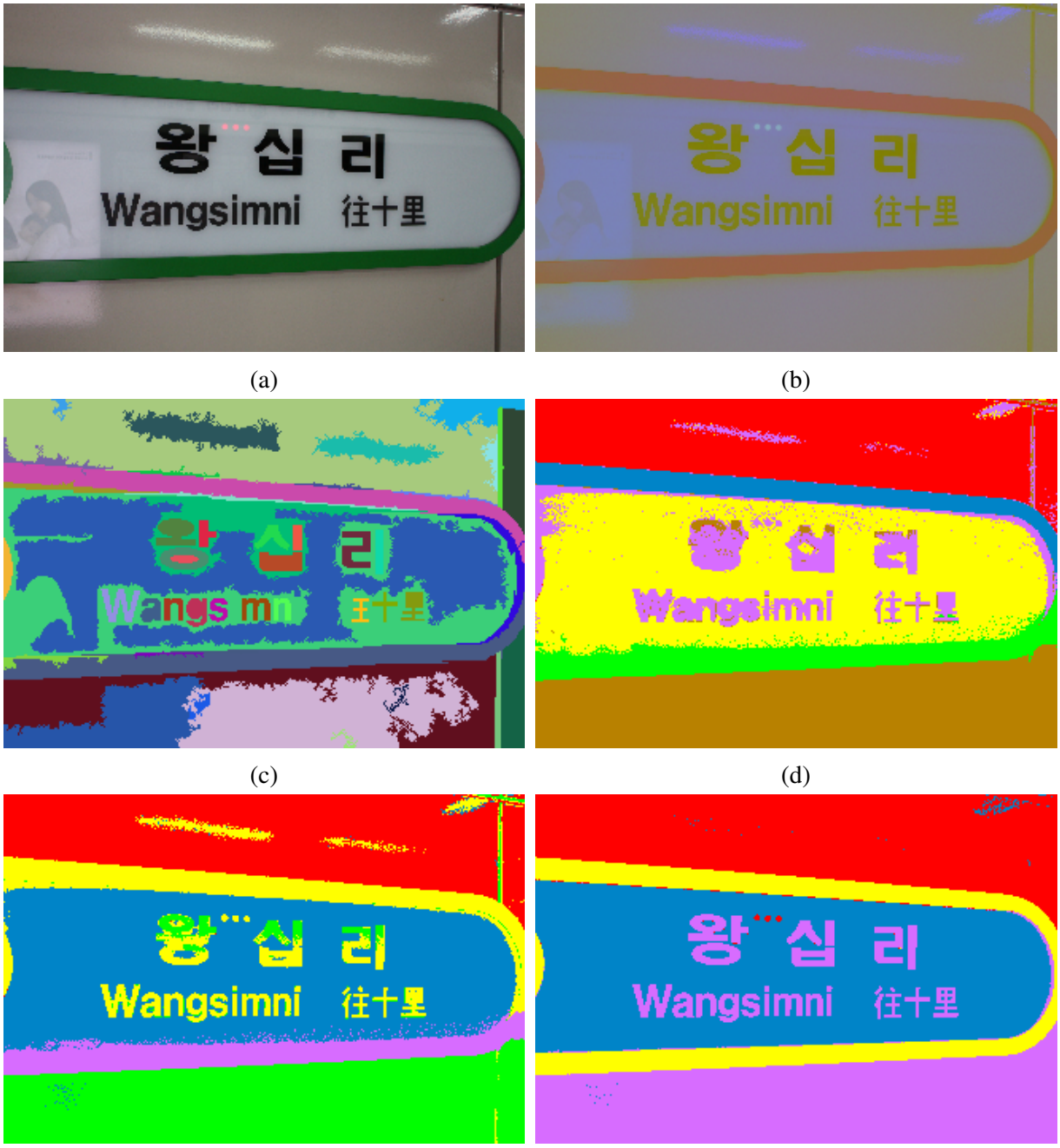


(a) (b)

(c) (d)

(e) (f)

Figure 4.5: Signboard fitting. Example 1. con't. a), c), e) separate signboards, and b, d), f) corresponding blob detection results.

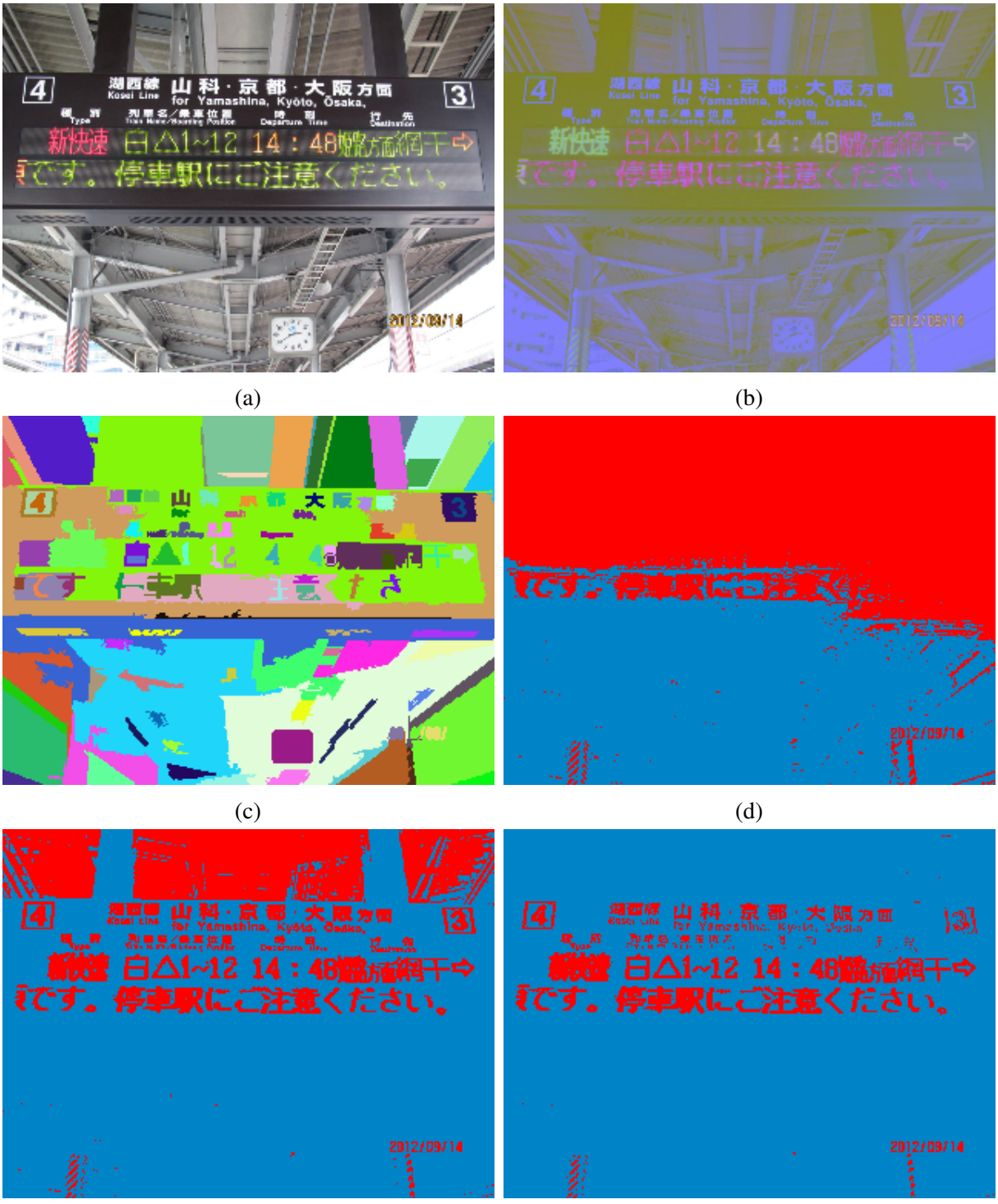


(a) (b)

(c) (d)

(e) (f)

Figure 4.6: Signboard fitting. Example 2. a) input image, b) image in Lab color space, c) super pixels, d) first iteration of signboard fitting, e) intermediate iteration and f) final result.

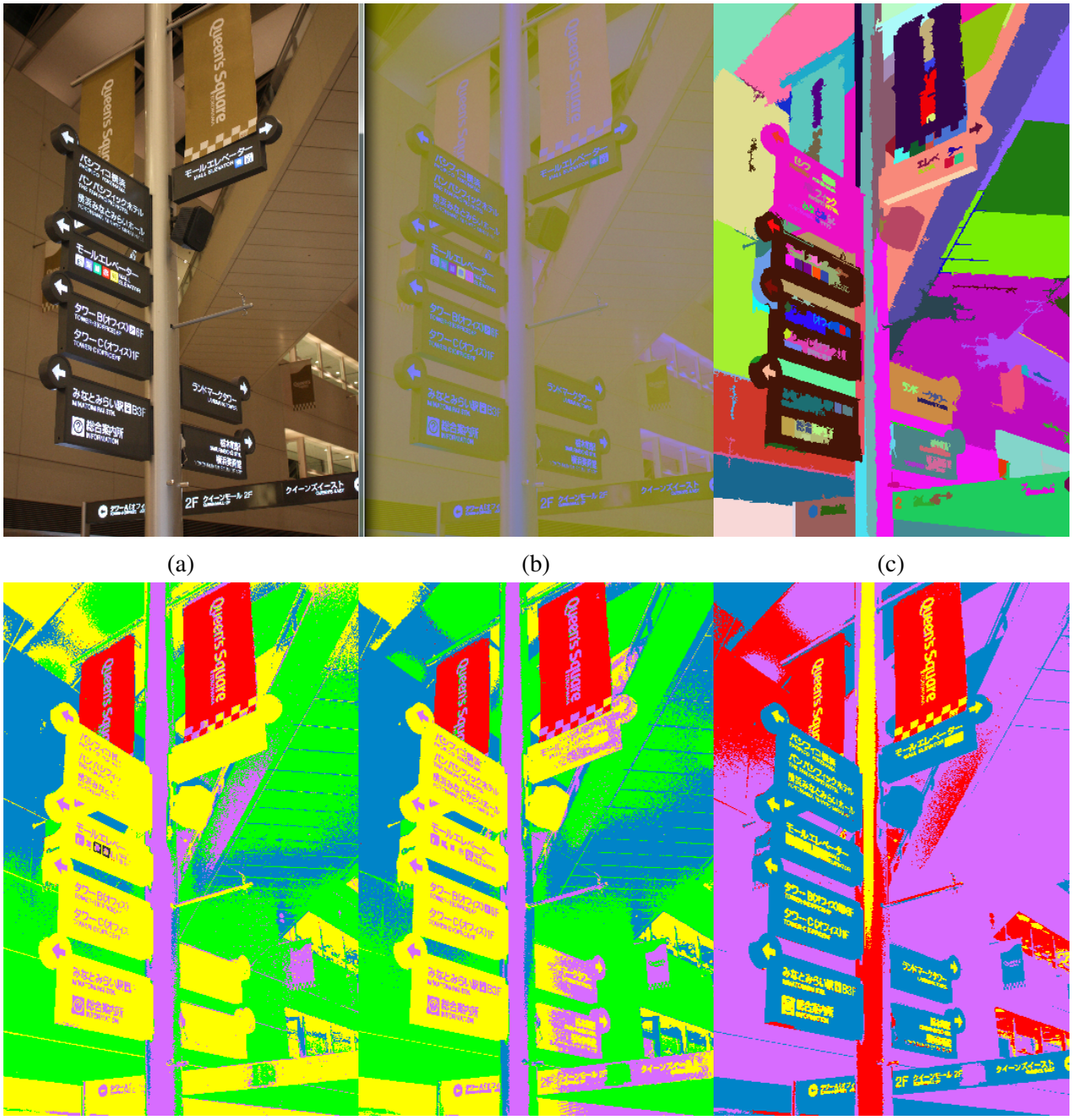


(a) (b)

(c) (d)

(e) (f)

Figure 4.7: Signboard fitting. Example 3. a) input image, b) image in Lab color space, c) super pixels, d) first iteration of signboard fitting, e) intermediate iteration and f) final result.



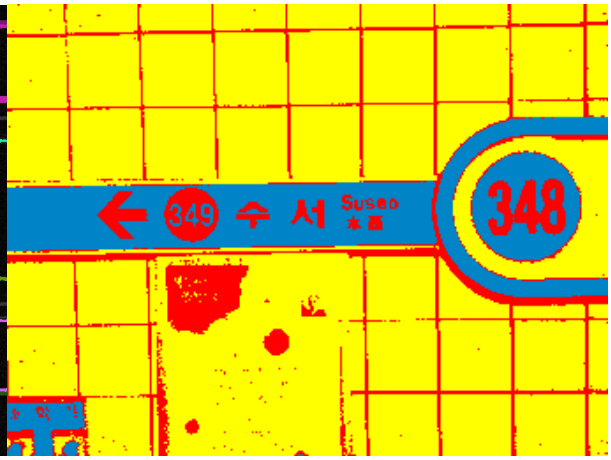
(a) (b) (c)

(d) (e) (f)

Figure 4.8: Signboard fitting. Example 4. a) input image, b) image in Lab color space, c) super pixels, d) first iteration of signboard fitting, e) intermediate iteration and f) final result.

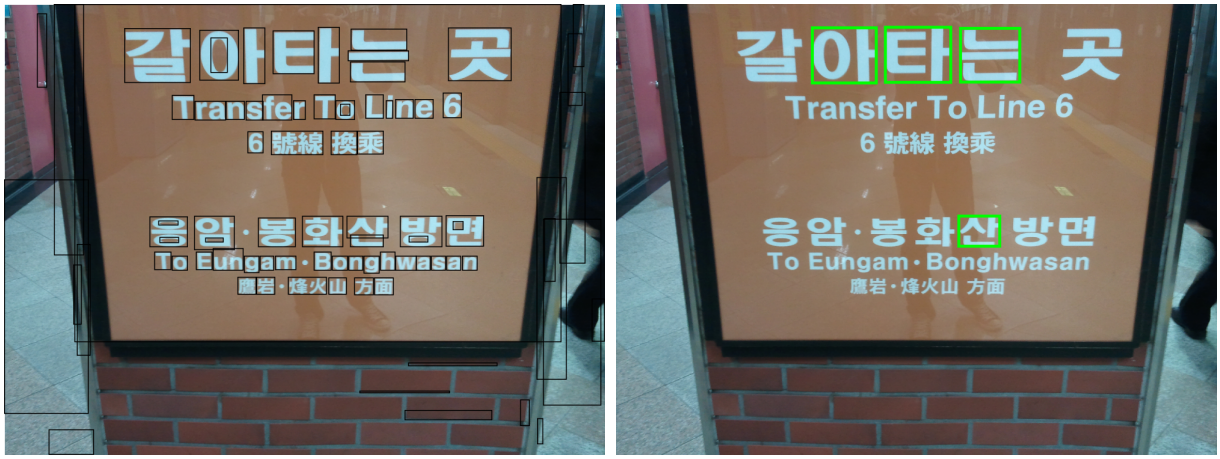
(a)

(b)



(c)

Figure 4.9: Edge-based vs. Color-based blob detection. a) Input image. b) Result of edge-based blob detection. Text blobs are missed due to weak edges. c) Result of color-based detection. All text blobs are found.



(a) (b)

Figure 4.10: Blob proposals. a) Blobs detected in the image. b) Artificial blobs added.

Chapter 5

Classification

5.1 Classifier

AdaBoost have become a popular choice for text vs. not-text recognition. AdaBoost is a machine learning algorithm that builds a strong classifier out of weak ones. In our problem we use a multi-class AdaBoost, which is trained on the following categories: English, Korean, Chinese, Digit, Non-Text. Decision trees are commonly used as a weak classifier. It is possible to scale the classifier according to the complexity of the problem by adjusting the depth and the quantity of the trees.

A blob that is marked as “text“ must cover part of a text line of a certain language. If it covers two or more lines (case of under-segmentation) or a part of a character (case of over-segmentation), then it is treated as non-text.

5.2 Features

The following groups of features are used for AdaBoost classifier training: color, gradient and geometry based features.

A bounding box that contains text has normally two Gaussians in a histogram of intensities; one for foreground and one for background. Non-text blobs do not have such consistency. For

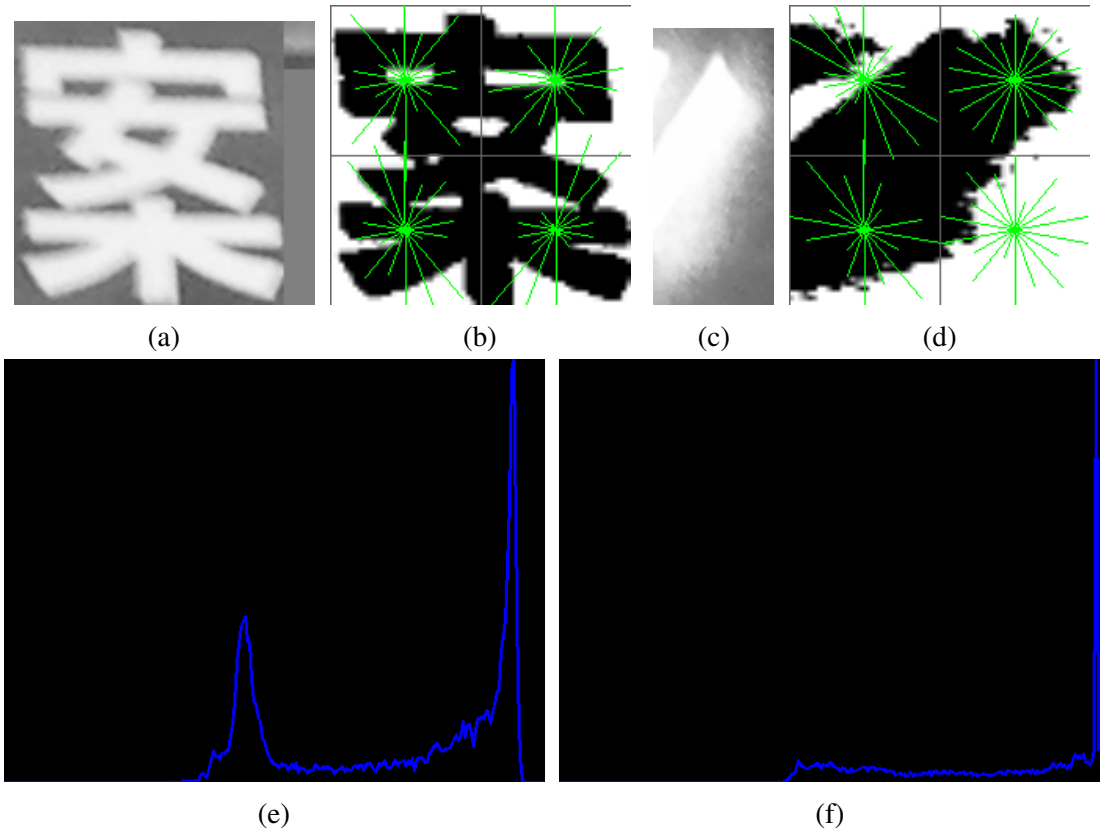


Figure 5.1: Color histograms and histograms of oriented gradients. Detected blob, HOG feature visualization and intensity histogram for Chinese character a) b) and e) and non-text blob c) d) and f) respectively.

instance, in Fig. 5.2 histogram of intensities of a Chinese character a) has two peaks, histogram of a non-text blob c) - one peak f). Histogram of intensities descriptor is computed as in [16].

Text of a different language could be diversified according to stroke density and number of strokes in a particular direction. We use the histogram of oriented gradients (HOG)[18] and Gabor fiter to describe such properties. For more details about Gabor filter see Appendix A.3.

Geometrical features of a blob are width, height and width-to-height ratio.

An example of text classification is shown in Fig. 5.3. Boxes in red, green, yellow and cyan denote English, Korean, Chinese characters and digits respectively. Non-text blobs are hidden for readability.

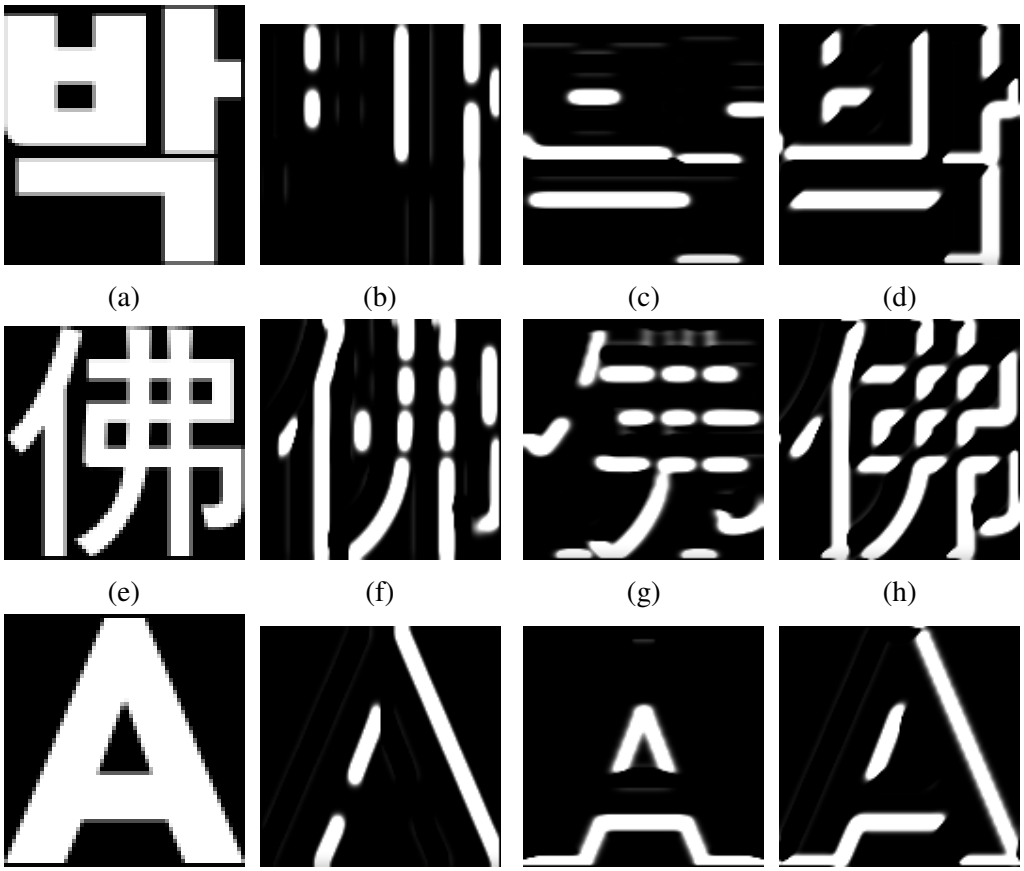


Figure 5.2: Gabor features. a), e) i) Characters in Korean, Chinese and English. b)-d), f)-h), j)-l) Gabor filtering in various orientations.

(a) (b)

(c) (d)

(e) (f)

Figure 5.3: Text candidate classification results. Non-text blobs are hidden. Red, green, yellow and cyan rectangle denote English, Korean, Chinese characters and digits respectively.

Chapter 6

Text line fitting

6.1 Objective function and optimization details

Assume that $\mathcal{I}$ is the set of all blobs i in the image, $\mathcal{L}$ is a set of indexes enumerating all possible text line models, and $\mathcal{V}$ is a set of indexes enumerating all possible languages. Vector $\boldsymbol{l}$ defines indexes of models assigned to all data points Eq (2.1). Vector $\boldsymbol{\theta}$ defines parameters for all models Eq (2.2). Text line model parameters θ are language marks and parameters of two non-parallel lines.

6.1.1 Objective function

The hierarchical objective function for text line detection could be formulated as

$$E(\boldsymbol{l}, \boldsymbol{\theta}) = \sum_{i \in \mathcal{I}} D_i(l_i|\boldsymbol{\theta}) + \sum_{j \in \mathcal{L}} C_j \cdot [\exists l_i = j] + \sum_{v \in \mathcal{V}} C_v \cdot [\exists v_{l_i} = v], \tag{6.1}$$

where $D_i(j|\boldsymbol{\theta})$ is the data error term describing how well each blob i fits text line j

$$D_i(j|\boldsymbol{\theta}) = -lnPr(i|language_j) + dist_{lang(j)}(i, lines_j). \tag{6.2}$$

It consist of two values: the AdaBoost classification score and geometrical error. The geometrical error is scaled according to the language of the model. The outlier cost $D_i(\varnothing) = const$, Constants C_j and is C_v a fixed penalties for each text line and each language in the solution respectively. Penalty C_j is incurred if there is at least one blob i assigned to text line j. In the same manner, penalty C_v is incurred if there is at least one blob i assigned to language v. The energy Eq. (6.1) describes the following hierarchical levels: *blobs→lines→languages*.

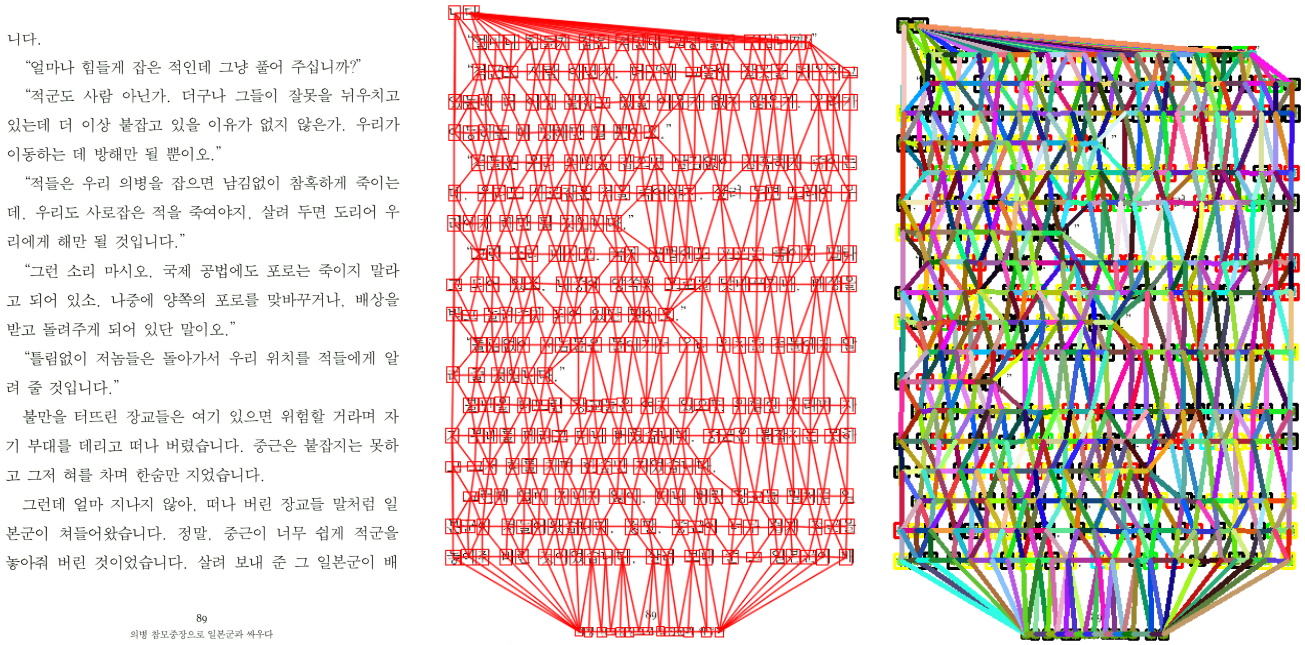
A text line fitting Algorithm's 1 input is a set of text candidates (blobs). The algorithm outputs the set of text line models and labeling. Labeling is a map that links the text candidates and the text lines.

A text line model is a pair of lines and a fixed language mark. Lines from one model are not necessarily parallel. The model can describe perspectively distorted or arbitrary rotated text lines of a certain language.

The Algorithm's 1 subroutines in application to the text line detection problem are define below. We propose the initial set of models as following. First, a neighborhood map is build by Delaunay triangulation. Next, blobs that have a common edge produce a model. Unlike k-nearest neighbor, such an approach enables a pair of blobs to produce a text line model, regardless of the distance between the blobs. One blob has a chance to produce text line models going in any directions in the rage of $[0, 2\pi)$. An example of model proposal for a page of printed text is shown in Fig. 6.1.

The model assignment links each blob to a model from the pool. On the model refit step the labeling is fixed. One model is fit to the blobs that have same label. In other words, only the data term of the energy Eq. (6.1) is minimized. The top and the bottom lines of a single model are refit by linear least squares (LS) (See more details on LS algorithm in Appendix A.1).

The model assignment Algorithm 2 is described as following. The models are fixed. A blob must be assigned to one of the models. First, initial labeling is made. All blobs are either assigned to the outlier label or the first model from the pool. The decision is made by thresholding. Next, the current solution l^0 is fused with labeling l^1 that again has one model



(a) (b) (c)

Figure 6.1: Text line proposal. a) input image b) Delaunay neighborhood map. c). Initial pool of text line models.

and the outlier. The fusion is a binary optimization technique that merges two solutions l^1 and l^0, while preserving the best qualities of each. The crossover operation is described by binary vector $\boldsymbol{x}$

$$\boldsymbol{x} = \{x_i | i \in \mathcal{I}\}, \tag{6.3}$$

where x_i is a binary variable. A blob i can be assigned to the model from either labeling l^0 or labeling l^1, i.e. $x \in \{0, 1\}$

$$l_i(x_i) = (1 - x_i)l_i^0 + x_i l_i^1. \tag{6.4}$$

When two labellings l^0 and l^1, and a pool of models with parameters $\boldsymbol{\theta}$ are given Eq. (6.1)

Algorithm 2: AssignModels

Input: *blobs* // text candidates
models // text lines
Output: *labeling* // blob-to-model map

```
l^0 ← MakeLabeling(∅, pool[0], blobs);
for i ← 1 to n_models do
    l^1 ← MakeLabeling(∅, pool[i], blobs);
    l^0 ← FuseLabeling(l^0, l^1);
labeling ← l_0
return labeling;
```

becomes

$$E(\boldsymbol{x}|l^0, l^1, \boldsymbol{\theta}) = \sum_{i\in\mathcal{I}} D_i^0 + (D_i^1 - D_i^0)x_i + \sum_{j\in\mathcal{L}} C_j \cdot (1 - \mathrm{X}_{P_j^0}\overline{\mathrm{X}}_{P_j^1}) + \sum_{v\in\mathcal{V}} C_v \cdot (1 - \mathrm{X}_{P_v^0}\overline{\mathrm{X}}_{P_v^1}), \tag{6.5}$$

where $X_{P_i^0}$ is an algebraic expression. It works as a switch. The switch is "on" (equals to 0) if there is at least one blob p assigned to the text line i from labeling 0, $\overline{X}_{P_i^1}$ is "on" if there is at least one blob p assigned to the text line i from labeling 1.

$$X_{P_i^0} = \prod_{p\in P_i^0} x_p = 0 \iff \exists p \in P_i^0 \tag{6.6}$$
$$\overline{X}_{P_i^1} = \prod_{p\in P_i^1} \bar{x}_p = 0 \iff \exists p \in P_i^1$$

$\mathrm{X}_{P_v^0}\overline{\mathrm{X}}_{P_v^1}$ is defined in alike manner.

Labeling fusion gives the optimal vector $\boldsymbol{x}$, which minimizes energy 6.5. The fusion is done by the graph cut algorithm[1].

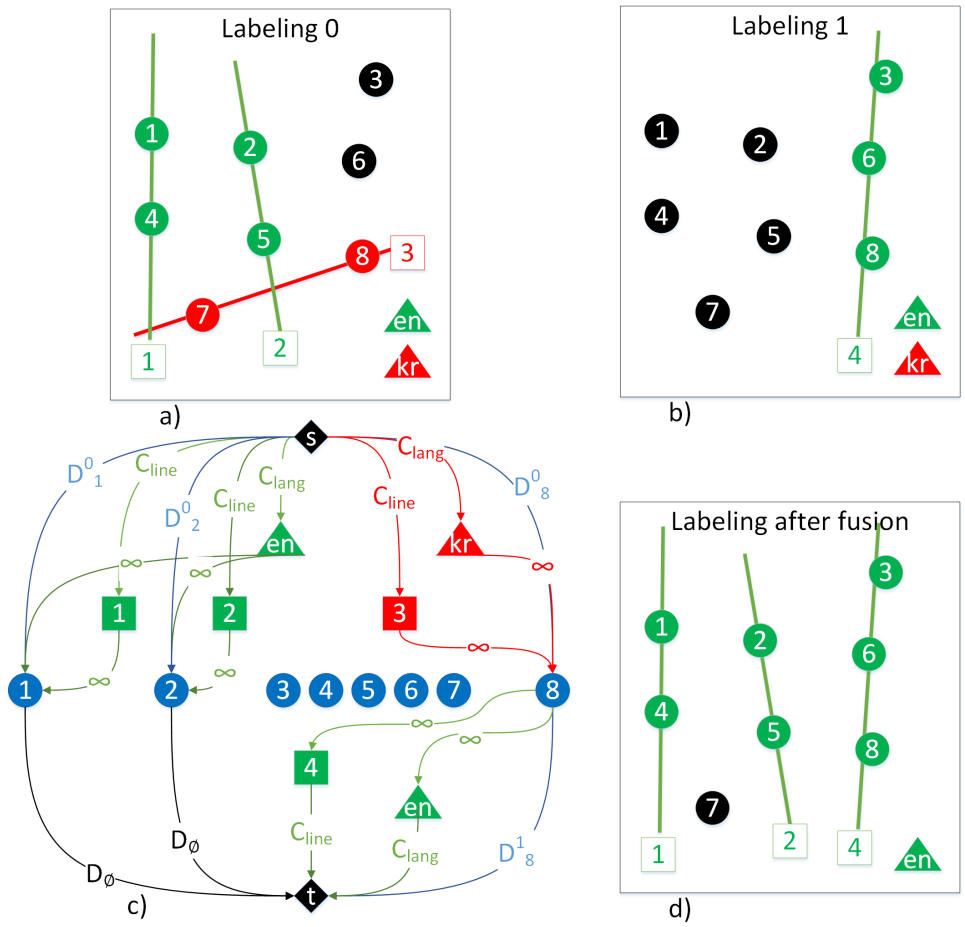


Figure 6.2: Labeling fusion. a) and b) show two labellings that will be fused. c) Graph, that represents fusion. The cuts of that graph define which points will take a label from labeling 0 or labeling 1. Edges from and to points ##3,4,5,6, and 7 are hidden, the models are shown as single lines for readability and d) Result of fusion. Point #7 is assigned to the outlier label.

6.2 Labeling fusion

An example of labeling fusion is shown in Fig. 6.2. There are eight blobs (circles). Labeling Fig. 6.2 a) has three lines, marked with squares. Labeling Fig. 6.2 b) has one line. Labeling l^0 Fig. 6.2 a) is:

$$l^0 = \{(1, en), (2, en), \varnothing, (1, en), (2, en), \varnothing, (3, kr), (3, kr)\}, \tag{6.7}$$

where points #3 and #6 are assigned to the outlier. Labeling l^1(Fig. 6.2 b) is:

$$l^1 = \{\varnothing, \varnothing, (4, en), \varnothing, \varnothing, (4, en), \varnothing, (4, en)\}, \tag{6.8}$$

where points ##1,2,4,5, and 7 are assigned to the oulier as well. Fig. 6.2 c) shows the graph that is used for optimization. A point can take model from labeling 0 or 1. The assignment of the point #1 to the model from labeling 0 will cost data term D_0^1. The assignment of the point #1 to the model from labeling 1 will cost a constant value - outlier cost. If a models is not connected to any point, then its cost will not be reflected in the energy Eq. (6.5). The minimum cut of the graph gives the vector $\boldsymbol{x}$ from Eq. (6.5). In our case $\boldsymbol{x} = (0, 0, 1, 0, 0, 1, 1, 1)$. Notice that this vector turns off the third line model. In the resulting labeling no points are assigned to this model and so $(1 - \mathrm{X}_{P^0}\overline{\mathrm{X}}_{P^1}) = (1 - x_7 \cdot x_8) = (1 - 1) = 0$. Line models 1,2 and 4 remain. The fusion produces new labeling (Fig.6.2 d):

$$l = \{(1, en), (2, en), (4, en), (1, en), (2, en), (4, en), \varnothing, (4, en)\}. \tag{6.9}$$

In the graph, incoming and outgoing links from and to points ##2,3,4 and 5 are hidden. The top half of the graph represents labeling 0, and the bottom half - labeling 1.

The following observations can be made about the energy formulation of the problem. One blob, which stands alone and does not fit any line, is too expensive. It must be assigned to the outlier label. A blob, which has a text likelihood score that is slightly lower than non-text score, still has a chance to be assigned to a line. The outlier model is necessary. It exists in each and every labeling and works as a garbage collector.

Chapter 7

Evaluation

7.1 Database

Our database of signboards from the Seoul subway consist of 500 images taken with a Samsung Galaxy SII, Galaxy S, Apple iPhone 3GS, Canon EOS 450D. 400 images were used for training and the remaining 100 for testing of both the AdaBoost classifier and the whole algorithm evaluation.

7.2 Classifier evaluation

The confusion matrix for text candidate classification by AdaBoost is shown in Table 7.1. The average recognition rate is 83.35 %.

Actual \ Predicted	English	Korean	Chinese	Digit	NonText
English	902	10	16	10	45
Korean	52	815	63	6	40
Chinese	27	14	172	0	11
Digit	19	1	0	121	5
NonText	482	305	188	144	5278

Table 7.1: Confusion matrix for classification.

7.3 Blob detector and line fitting evaluation

We want to test all three steps of the algorithm in an isolated fashion. However, it is not straightforward to check how well blobs that cover text are detected. Instead, we compare final detected line segments, found in one image, to two ground truths - one contains only blobs that were detected by the first stage of the algorithm (Artificial base), the second contains all text blobs placed there by a human (Real base). The difference between accuracy of these two experiments reflects the quality of the blob detector.

We estimated recall and precision metrics Eq. (7.1) and Eq. (7.2) that are commonly used for text detection evaluation [30, 38]. When a solution and a ground truth have more than one text line the following method is applied. Precision and recall are estimated for a line in the solution and all lines in the ground truth. Maximum precision and recall are selected. Both metrics represent interactions area of blobs from ground truth and text line are. Precision is normalized by the area of blobs from the solutions, recall - by the area of blobs from the ground truth. In other words precision tells what percentage of the text line are from ground truth was found, recall - what area of the found text line does not exist in the ground truth.

$$Precision = \frac{\{ground\ truth\ blobs\} \cap \{found\ blobs\}}{\{found\ blobs\}} \tag{7.1}$$

$$Recall = \frac{\{ground\ truth\ blobs\} \cap \{found\ blobs\}}{\{ground\ truth\ blobs\}}. \tag{7.2}$$

The performance of the text line detection is shown in Table 7.2.

Ground truth	Recall (%)	Precision (%)	F (%)
Real base	71	84	76
Artificial base	76	84	79

Table 7.2: Text line detection accuracy.

Examples of text line detection are shown in Fig. 7.1. A triplet of lines in one colour shows one line. Circles mark centres of text blobs.

(a) (b) (c) (d) (e) (f)

Figure 7.1: Results of text line detection. Triplet of lines in one colour shows one line. Circles mark centres of text blobs. Some blobs were misclassified in b),d),e) and f).

Chapter 8

Conclusion and future work

8.1 Summary

In this work we introduced a challenging new problem of the multilingual multi-line text detection. We formulated the problem as a hierarchical MDL energy optimization and demonstrated that a fusion based method efficiently obtains good quality solutions for this energy. We obtained very promising results on our large database of images from the subway of the metropolitan area of Seoul that we plan to make public for other researchers in computer vision.

Our energy for text line detection aims to describe the diversity of text candidates by the smallest number of text lines and languages. A text line describes text candidates in one language only. The last mentioned property makes the text recognition task simpler. Instead of using one complex recognizer that has to deal with all various characters in all languages at once, a set of unilingual recognizers could be used.

Moreover, we proposed a color-based energy minimization approach for signboard fitting and blob detection. Unlike an edge-based blob detector, our method works well when the camera failed to focus properly. Additionally, our method is more robust than existing algorithms in images with artifacts, i.e. shadows, glare, and reflections.

8.2 Future work

Our experiments showed that the bottle neck of our algorithm is the AdaBoost classifier performance, which could be enhanced. We plan to enlarge our training database with artificial samples and additional signboard images of the subway from the web. Extending our current hierarchy *blobs→lines→languages* into *blobs→ characters →lines→languages* could further improve the results.

We plan to replace currently used edge-based blob detector with our novel algorithm. The resulting text line detector can be applied to a wide range of problems in navigation, text translation, etc. It can be extended to other sets of languages, including Japanese, Russian, German, French, Hebrew and Thai.

Bibliography

[1] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

[2] A. B. Cambra and A.C. Murillo. Towards robust and efficient text sign reading from a mobile phone. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 64–71, 2011.

[3] Xilin Chen, Jie Yang, Jing Zhang, and Alex Waibel. Automatic detection and recognition of signs from natural scenes. *Image Processing, IEEE Transactions on*, 13(1):87–99, 2004.

[4] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, Tao Wang, D.J. Wu, and A.Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 440–445, 2011.

[5] Franklin C. Crow. Summed-area tables for texture mapping. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 207–212, New York, NY, USA, 1984. ACM.

[6] Andrew Delong, Anton Osokin, Hossam N. Isack, and Yuri Boykov. Fast approximate energy minimization with label costs. *Int. J. Comput. Vision*, 96(1):1–27, January 2012.

[7] Andrew Delong, Olga Veksler, Anton Osokin, and Yuri Boykov. Minimizing sparse high-order energies by submodular vertex-cover. In *NIPS*, pages 971–979, 2012.

[8] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963–2970, 2010.

[9] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, September 2004.

[10] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):855–868, 2009.

[11] J. He, Q. D M Do, A.C. Downton, and J.-H. Kim. A comparison of binarization methods for historical archive documents. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pages 538–542 Vol. 1, 2005.

[12] Hossam Isack and Yuri Boykov. Energy-based geometric multi-model fitting. *Int. J. Comput. Vision*, 97(2):123–147, April 2012.

[13] Sezer Karaoglu, Jan van Gemert, and Theo Gevers. Object reading: Text recognition for object recognition. 7585:456–465, 2012.

[14] Lubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip H. S. Torr. Graph cut based inference with co-occurrence statistics. In *Proceedings of the 11th European conference on Computer vision: Part V*, ECCV'10, pages 239–253, Berlin, Heidelberg, 2010. Springer-Verlag.

[15] L'ubor Ladický, Paul Sturgess, Karteek Alahari, Chris Russell, and Philip H. S. Torr. What, where and how many? combining object detectors and crfs. In *Proceedings of*

the 11th European conference on Computer vision: Part IV, ECCV'10, pages 424–437, Berlin, Heidelberg, 2010. Springer-Verlag.

[16] Jung-Jin Lee, Pyoung-Hean Lee, Seong-Whan Lee, A. Yuille, and C. Koch. Adaboost for text detection in natural scene. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 429–434, 2011.

[17] Hongdong Li. Two-view motion segmentation from linear programming relaxation. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.

[18] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[19] R. Minetto, N. Thome, M. Cord, J. Stolfi, F. Precioso, J. Guyomard, and N. J. Leite. Text detection and recognition in urban scenes. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 227–234, 2011.

[20] A. Mishra, K. Alahari, and C.V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2687–2694, 2012.

[21] L. Neumann and J. Matas. Text localization in real-world images using efficiently pruned exhaustive search. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 687–691, 2011.

[22] L. Neumann and J. Matas. Real-time scene text localization and recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3538–3545, 2012.

[23] Yi-Feng Pan, Xinwen Hou, and Cheng-Lin Liu. Text localization in natural scene images based on conditional random field. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 6–10, 2009.

[24] Yi-Feng Pan, Xinwen Hou, and Cheng-Lin Liu. A hybrid approach to detect and localize texts in natural scene images. *Image Processing, IEEE Transactions on*, 20(3):800–813, 2011.

[25] Yi-Feng Pan, Yuanping Zhu, Jun Sun, and S. Naoi. Improving scene text detection by scale-adaptive segmentation and weighted crf verification. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 759–763, 2011.

[26] M. Petter, V. Fragoso, M. Turk, and Charles Baur. Automatic text detection for mobile augmented reality translation. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 48–55, 2011.

[27] Trung Quy Phan, P. Shivakumara, and C.L. Tan. A laplacian method for video text detection. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 66–70, 2009.

[28] J. Porway, K. Wang, B. Yao, and Song-Chun Zhu. A hierarchical and contextual model for aerial image understanding. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[29] K. Schindler and D. Suter. Two-view multibody structure-and-motion with outliers. 2:643–648 vol. 2, 2005.

[30] A. Shahab, F. Shafait, and A. Dengel. Icdar 2011 robust reading competition challenge 2: Reading text in scene images. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1491–1496, 2011.

[31] O. Shiku, K. Kawasue, and A. Nakamura. A method for character string extraction using local and global segment crowdedness. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, volume 2, pages 1077–1080 vol.2, 1998.

[32] P. Shivakumara, Trung Quy Phan, and C.L. Tan. A gradient difference based technique for video text detection. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 156–160, 2009.

[33] Roberto Toldo and Andrea Fusiello. Robust multiple structures estimation with j-linkage. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 537–547, Berlin, Heidelberg, 2008. Springer-Verlag.

[34] P. H. S. Torr. Geometric motion segmentation and model selection. *Phil. Trans. Royal Society of London A*, 356:1321–1340, 1998.

[35] Wikipedia. Gabor filter, 2013. [Online; accessed 15-December-2013].

[36] Wikipedia. Integral image, 2013. [Online; accessed 15-December-2013].

[37] Wikipedia. Least squares, 2013. [Online; accessed 15-December-2013].

[38] C. Wolf and J.-M. Jolion. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *International Journal on Document Analysis and Recognition*, 8(4):280–296, 2006.

[39] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1083–1090, 2012.

[40] Yan Zhao, Tong Lu, and Wujun Liao. A robust color-independent text detection method from complex videos. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 374–378, 2011.

[41] M. Zuliani, C.S. Kenney, and B.S. Manjunath. The multiransac algorithm and its application to detect planar homographies. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–153–6, 2005.

Appendix A

Computer vision techniques

A.1 Least squares

The method of least squares is a standard approach to the finding the best-fitting line to a given set of points by minimizing the sum of the squares of the offsets of the points from the line. In other words, the goal of the least squares algorithm is for a given set of points $\boldsymbol{x} = \{(x, y)\}$ to find a line parameters $\theta = (A, B, C)$ Eq. (A.1), so that sum of the squared Euclidean distances from the points $\boldsymbol{x}$ to the line is minimal Eq. (A.2).

$$Ax + By + C = 0 \tag{A.1}$$

$$E(\theta) = \sum_{i \in \boldsymbol{x}} \left(\frac{|Ax_i + By_i + C|}{\sqrt{A^2 + B^2}} \right)^2 \tag{A.2}$$

Principal component analysis (PCA) can be used for linear least squares [37]. PCA takes in the vector $\boldsymbol{x}$ and outputs eigenvalues λ and eigenvectors v. In two dimensional case there are two eigenvalues and two eigenvectors. The eigenvector associated with the biggest eigenvalue is the first principal component (Fig. A.1). Knowing the first principal component (eigenvector (v_x, v_y) and data's mean μ_x and μ_y model parameters minimizing energy Eq. (A.2) are:

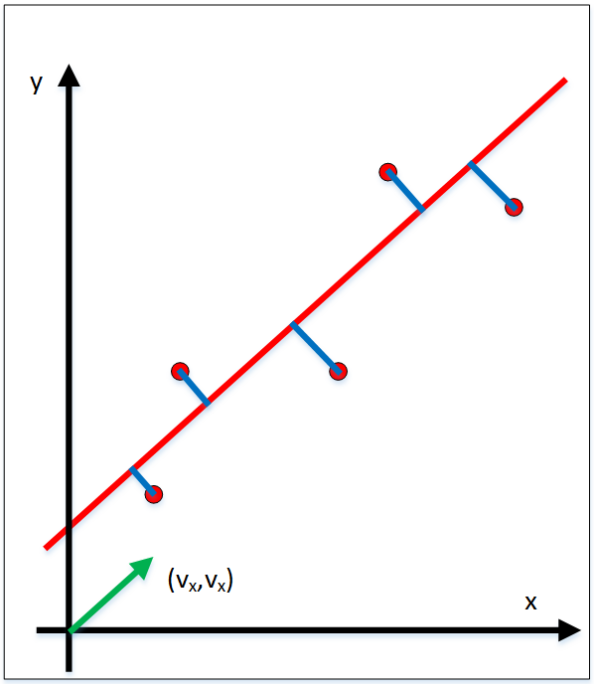


Figure A.1: Linear least squares problem. The red dots are the data points, the red line is the optimal line model. The blue line segments show euclidean distances from the data points to the line. The green vector is the first principal component.

$$A = v_x \tag{A.3}$$

$$B = v_y$$

$$C = -(A \cdot \mu_x + B \cdot \mu_y)$$

A.2 Integral image

A integral image is an algorithm for an effective sum of values in a rectangular subset of a image [5, 36]. The value at any point (x, y) in the integral image is the summation of all the

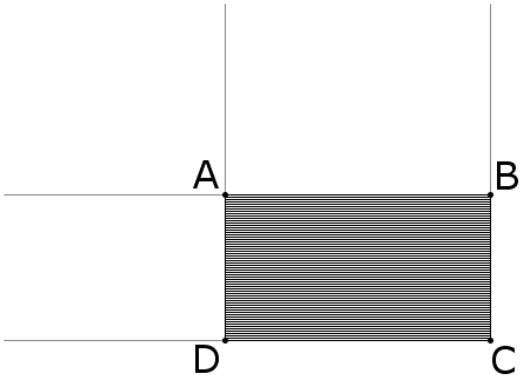


Figure A.2: Finding the sum of a rectangular area using integral image.

pixels above and to the left of (x, y), inclusive:

$$I(x, y) = \sum_{\substack{x' \leq x \\ y' \leq y}} i(x', y') \tag{A.4}$$

An integral image can be computed efficiently in a single pass over the image:

$$I(x, y) = i(x, y) + I(x - 1, y) + I(x, y - 1) - I(x - 1, y - 1), \tag{A.5}$$

Once the integral image is computed, the task of evaluating any rectangle can be accomplished in constant time with four array references. The sum of intensities i(x,y) over the rectangle spanned by A, B,C and D is:

$$\sum_{\substack{x0 \leq x \leq x1 \\ y0 \leq y \leq y1}} i(x, y) = I(C) + I(A) - I(B) - I(D), \tag{A.6}$$

where $A = (x_0, y_1)$, $B = (x_1, y_1)$, $C = (x_1, y_0)$ and $D = (x_0, y_0)$ (Fig. A.2). After the integral image is obtained the computation of μ (mean intensity inside a window around a pixel (x, y))

can be done in constant time:

$$\mu = \frac{I(C) + I(A) - I(B) - I(D)}{(x_1 - x_0)(y_1 - y_0)}. \tag{A.7}$$

A.3 Gabor filter

Gabor filter is a linear filter used for edge detection, data compression, face recognition, texture analysis, handwriting recognition and other image processing problems [3, 35]. For a given pixel (x_1, y_1) with intensity $I(x_1, y_1)$ in an image, its Gabor feature is the result of convolution:

$$J(x_1, y_1) = \iint I(x_1 - x, y_1 - y)\, g(x, y; \lambda, \theta, \psi, \sigma, \gamma)\, dxdy, \tag{A.8}$$

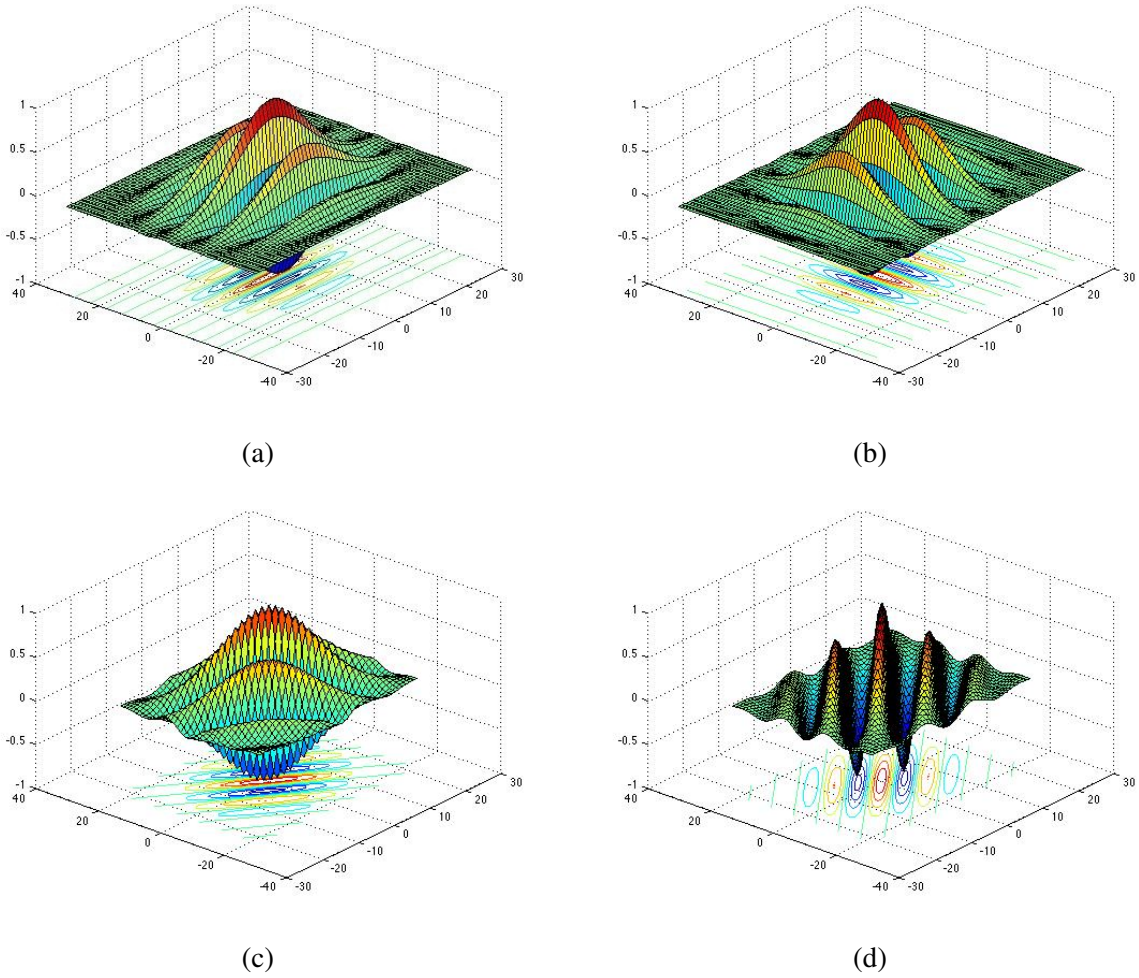
where Gabor kenerl $g(x, y; \lambda, \theta, \psi, \sigma, \gamma)$ is :

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \sin\left(2\pi \frac{x'}{\lambda} + \psi\right), \tag{A.9}$$

$$x' = x\cos\theta + y\sin\theta, \tag{A.10}$$

$$y' = -x\sin\theta + y\cos\theta, \tag{A.11}$$

and σ is the deviation of the Gaussian envelope, ψ is the phase offset, γ is the spatial aspect ratio, λ and θ are the wavelength and the orientation of the Gabor function respectively. Gabor kernels in four orientations (0 , 45 , 90 , and 135) are shown in Fig. A.3.



(a) (b)

(c) (d)

Figure A.3: Gabor filter kernels in four orientations: a) $\theta = 0°$, b) $\theta = 90°$, c) $\theta = 45°$ and d) $\theta = 135°$

Curriculum Vitae

Name: Igor Milevskiy

Education:

Masters in Computer Science, University of Western Ontario

Advisor: Professor Yuri Boykov

Thesis: *Detecting multilingual lines of text with fusion moves*

Sept. 2012 Dec. 2013(expected)

Masters in Computer Science, Kangwon National University

Advisor: Professor Jin-Young Ha

Thesis: *Subway Path Finding using Character Recognition on Smart Phones*

Aug. 2009 Aug. 2011

International Exchange Program, Changwon National University

Aug. 2007 June 2008

Specialist Degree in Computer Science, Pacific National University

Advisor: Professor Robert V. Namm

Thesis: *Approximate Solution of Half-Coercive Model Problem with Friction by an Iterative Proximal Regularization*

Sept. 2004 June 2009

Related Work Experience:

Teaching Assistant
The University of Western Ontario
2012 - 2013

Teaching Assistant
Kangwon National University
2010

Honours and Awards:

WGRS Western Graduate Research Scholarship
2012 Present

Graduate Student Scholarship
2012 Present

The IBM - Western University Industry Problem Solving Workshop
Data Analytics Problem Solving (DAPS) Trophy
Feb. 2013

ACES-KNU Academic Counterparts Elites Scholarship
Scholarship for graduate studies at Kangwon National University
2009 2011

Changwon National University Scholarship for Exchange Studies
2007 2008

Russian Government Scholarship for Undergraduate Studies
2004 2009

Publications: Igor Milevskiy and Jin-Young Ha. A fast algorithm for Korean text extraction and segmentation from subway signboard images utilizing smartphone sensors. *JCSE*, 5(3):161–166, 2011.