# A CONSTRAINT BASED REAL-TIME LICENSE PLATE RECOGNITION SYSTEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ALİ GÖKAY GÜNAYDIN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

FEBRUARY 2007

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Ayşe Kiper
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Adnan Yazıcı
Supervisor

**Examining Committee Members**

| | | |
|---|---|---|
| Assoc. Prof. Dr. İ. Hakkı Toroslu | (METU, CENG) | |
| Prof. Dr. Adnan Yazıcı | (METU, CENG) | |
| Assoc. Prof. Dr. Ali Doğru | (METU, CENG) | |
| Assoc. Prof. Dr. Ferda Nur Alpaslan | (METU, CENG) | |
| Arslan Arslan | (LOGO Yazılım) | |

# ABSTRACT

A CONSTRAINT BASED REAL-TIME
LICENSE PLATE RECOGNITION SYSTEM

Günaydın, Ali Gökay
M.Sc., Department of Computer Engineering
Supervisor : Prof. Dr. Adnan Yazıcı

February 2007, 48 pages

License Plate Recognition (LPR) systems are frequently utilized in various access controls and security applications. In this thesis, an experimental constraint based real-time License Plate Recognition system is designed, and implemented in Java platform. Many of the available constraint based methods worked under strict restrictions such as plate color, fixed illumination and designated routes, whereas, only the license plate geometry and format constraints are used in this developed system. These constraints are built on top of the current Turkish license plate regulations. The plate localization algorithm is based on vertical edge features where constraints are used to filter out non-text regions. Vertical and horizontal projections are used for character segmentation and Multi Layered Perceptron (MLP) based Optical Character Recognition (OCR) module has been implemented for character identification. The extracted license plate characters are validated against possible license plate formats during the recognition process. The system is tested both with Turkish and foreign license plate images including various plate orientation, image quality and size. An accuracy of 92% is achieved for license plate localization and %88 for character segmentation and recognition.

Keywords: License Plate Recognition, Text Extraction, Image Enhancement, OCR

# ÖZ

KISIT TABANLI GERÇEK ZAMANLI PLAKA TANIMA SİSTEMİ

Günaydın, Ali Gökay
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Adnan Yazıcı

Şubat 2007, 48 sayfa

Plaka Tanıma Sistemleri sıklıkla erişim kontrol ve güvenlik uygulamalarında kullanılır. Bu tez içeriside, kısıt tabanlı gerçek zamanlı bir Plaka Tanıma Sistemi tasarlanmış ve Java platformu üzerinde geliştirilmiştir. Önceki çalışmaların birçoğu plaka rengi, sabit aydınlatma koşulları, belirli rotalar gibi katı kısıtlar üzerinden çalışırken, geliştirilen sistemde yalnızca plaka boyutları ve genel düzen kısıtları kullanımıştır. Bu kısıtlar Türkiye'deki plaka düzenlemesi üzerine inşa edilmiştir. Plaka yerini belirleme algoritması, kısıtların dikey köşe özellikleri üzerinden metin olmayan alanların filtrelenmesi üzerine kurulmuştur. Karakter ayrıştırma işlemi için yatay ve dikey izdüşümleri, Optik Karakter Tanıma modülü (OCR) için ise Çok Katmanlı Algılayıcılar (MLP) kullanılmıştır. Elde edilen plaka karakterlerinin olası plaka düzenlerine göre uygunluğu tanıma işlemi sırasında konrol edilir. Önerilen sistem hem Türk hem de yabancı kaynaklı, görüntü kalitesi ve büyüklüklerine göre çeşitlilik gösteren plakalar ile denemiştir. Çalışmada plaka yerinin tespti için %92, karakter ayrıştırma ve tanıma için %88 doğruluk oranlarına ulaşılmıştır.

Anahtar Kelimeler: Plaka Tanıma Sistemi, Metin Çıkarma, Görüntü İyileştirme, Optik Karakter Tanıma

To My Family

# ACKNOWLEDGMENTS

The author wishes to express his deepest gratitude to his supervisor Prof. Dr. Adnan Yazıcı for his guidance, advice, criticism and encouragements throughout the research.

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

License Plate Recognition (LPR) is a combination of image processing, character segmentation and recognition technologies used to identify vehicles by their license plates. Since only the license plate information is used for identification, this technology requires no additional hardware to be installed on vehicles. LPR technology is constantly gaining popularity, especially in security and traffic control systems [16]. License Plate Recognition Systems are utilized frequently for access control in buildings and parking areas, law enforcement, stolen car detection, traffic control, automatic toll collection and marketing research. There are many successful commercial systems [16] available; however, there exists little documentation and public information about LPR system internals like the algorithms used in plate localization and recognition.

License plate recognition is a part of a more general research area called Text Information Extraction (TIE) [32, 33]. TIE algorithms are used to extract textual information from video streams and images. In the LPR problem, the textual information is the license plate characters. Similar to other TIE applications, license plate recognition involves localization, extraction, enhancement and recognition of characters in a given image. However, unlike applications like document recognition [39, 42], LPR systems generally operate on noisy and low quality images, in which illumination conditions may frequently cause difficulties.

LPR applications apply image processing and segmentation algorithms for license plate extraction, and each operation involves lots of computation. Government regulations and standards employed in the license plates can reduce the computational requirements substantially and improve the accuracy. Constraints and value ranges can be built on top of this priori knowledge, and used for extraction as well as validation of license plates. Constraints contain range of values instead of exact measures, since the license plate text size, style and orientation can vary substantially in different images.

## 1.1 Introduction to License Plates

License plates have different size, character format, base material and color standards throughout the world. Figure 1 illustrates example plate images from different countries. Generally, license plates are characterized by high contrast between characters and underlying background. However, license plates in some countries may contain background texture and images, which introduces extra complexity in localization and extraction of license plate information.

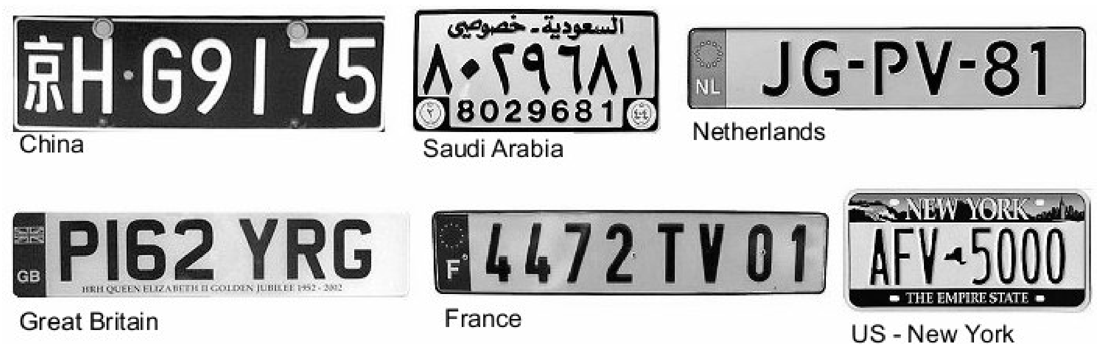


**Figure 1 Sample license plate images [37]**

In some countries, different plate standards are used simultaneously. Figure 2 shows examples of different license plate standards currently used in Turkey. The differences in font, size, style and colors used in license plates are remarkable.

Turkish license plates are rectangular in shape and made of aluminum. Current regulations impose standards on the issued license plates; however, previously acquired license plates may differ in size and style. In the present format, there is the country code "TR" on the left in a 4x10cm blue bar like in European

Union countries. The text is in black characters on white background, and for official vehicles white on black. Foreground colors like green, blue and red as well as background color yellow are also used on special purpose license plates, but rarely encountered.

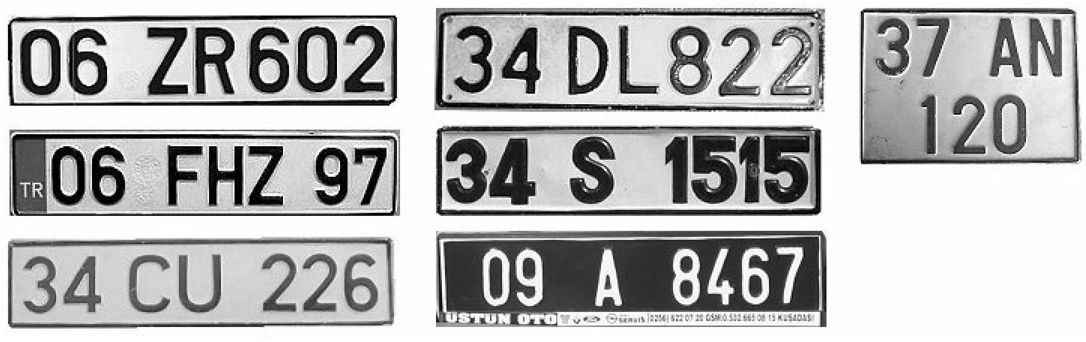


**Figure 2 Different Turkish license plates [37]**

On all vehicles, two plates have to be present, being one in front and the other in rear. The size of the plate is;

- 15x24cm in front and rear for motorbikes, motorcycles and tractors with rubber wheels
- 11x52cm in front and 21x32cm rear for cars, pickups, vans, trucks and busses.
- 15x30cm for imported vehicles if the regular plate does not fit.

The text format on the license plates is one of the following;

- DD L DDDD or DD L DDDDD
- DD LL DDD or DD LL DDDD
- DD LLL DD or DDD LLL DDD

The character ‘D’ denotes digits and ‘L’ represents letters in the character formats above. Each group of character is delimited by a space, yet the size of the space can vary from one plate to another. The first two digits denote the location, and there are currently 81 provinces in Turkey. In addition, special license plate formats exist for military and governmental vehicles; however, these formats are not included in building the LPR constraints.

## 1.2 Introduction to LPR Systems

Recent improvements in technology like infrared imaging and high resolution cameras, and utilization of high reflective backgrounds in license plate manufacturing have improved the accuracy of LPR systems. Sensors and other hardware peripherals are used to improve the image acquisition and remove irrelevant details. New video frame grabbers enable fast digitization of acquired images and new computers with proper LPR software render the system operable in real-time with high precision and accuracy. After the recognition process, the extracted information can be further verified through local or remote databases and stored for future referencing. A typical LPR system is composed of several hardware and software components as illustrated in Figure 3.

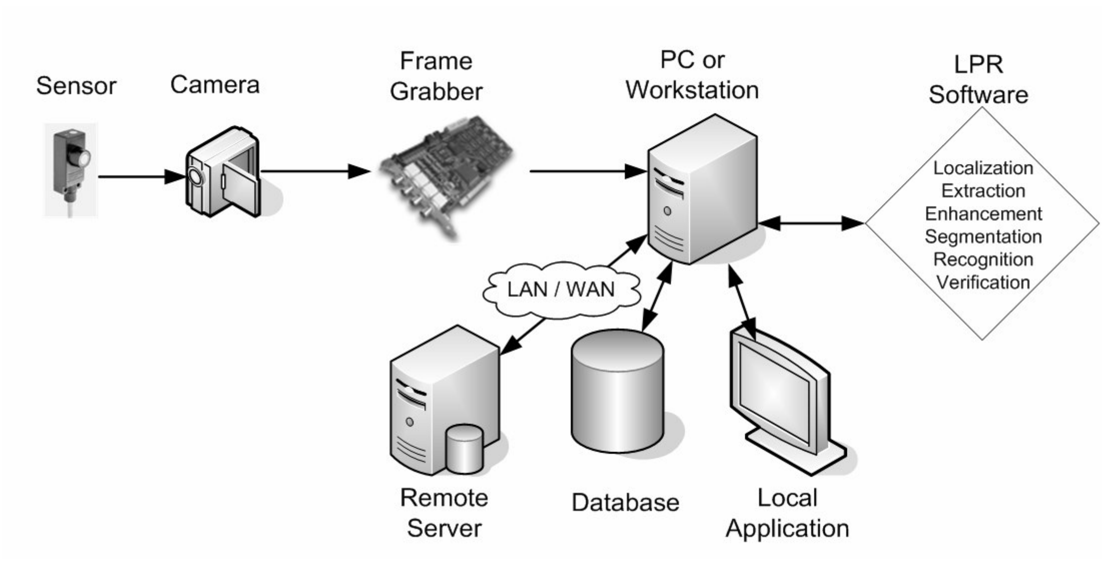


**Figure 3 A typical LPR system**

A typical LPR system operates as follows. When the vehicle approaches the secured area or gate, it starts the cycle by stepping over a magnetic loop detector (which is the most popular vehicle sensor). The loop detector senses the car and sends signals to the LPR unit. The LPR unit activates the illumination (invisible Infra-red in most cases) and takes pictures of the front or rear plates from the LPR camera. The images of the vehicle include the plate and the pixel information is read by the LPR unit's image processing hardware (the frame grabber). The LPR unit analyzes the image with different image processing software algorithms, enhances the image, detects the plate position, extracts the plate string, and identifies the fonts using special artificial intelligence methods (such as Neural Networks). The extracted license plate information can be logged, stored along with the captured image or used for authentication depending on the LPR application.

The recognition is not absolute and may contain errors due to problems in any of the LPR stages. Applications need to employ proper verification and control methods in order to compensate for the potential problems. If the application is sensitive to errors (e.g. toll collection), license plates that are not correctly identified are processed manually.

## 1.3 Problem Definition

Automatic recognition of license plates requires several image processing and computer vision algorithms to be utilized within a single application. Text localization, extraction and enhancement, character segmentation and recognition operations are used to determine the license plate number in a given image or video frame. Only a few of the previous studies involve all the steps of a typical LPR system, from image acquisition to verification. In this research, a complete license plate recognition system, which is based on constraints and operates in real-time, has been designed and implemented.

License plate localization and extraction are the most time consuming stages of a typical LPR system. Assumptions as  well as optimizations are required in order for LPR systems to be able to locate license plates in real time. Previous studies used specific features like background color [5, 20], plate border [3], size and symmetry [24], shape [6] and texture [4]. Furthermore, regular intervals between plate character [18] and near-uniform background intensity [8] and the sign transition of gradient [38] are also used for plate localization. As more features are utilized in LPR systems, the overall accuracy of the localization increases. However, the computational requirements increase in parallel. To minimize this side-effect, constraints and priori knowledge are utilized. In the proposed system, a combination of vertical edge features, sign transitions of gradient and near-uniform background intensity properties are used in accordance with government regulations.

After extracting the license plate area, the resulting region is further processed for character segmentation and recognition. The former has been studied in the past and many successful methods like projection [3], morphology [4], connected components [20], coloring and labeling have been proposed. The projection method is simple and effective, however it assumes the orientation of a license plate is known. The morphology method requires knowing the sizes of characters, which can only be estimated roughly. The connected component approach assumes that the characters touch neither to each other nor to the plate borders. As the quality of license plate images are not high enough, connected component method often results in character splits or merges. Coloring and labeling methods involve iterations, and often they are not suitable due to time constraints. Since the license plate image can be corrected for rotation, the projection method is used in the proposed system for the best performance.

Segmented characters are fed into a recognition engine for final identification. Several open source [40] and commercial OCR [39, 41] systems have been utilized during experiments. The recognition results were very poor since the image quality and resolution were not adequate for these kinds of products. The best hits contained only three characters of the extracted license plate image, which was not adequate for LPR applications. Although implementing an optical character recognition (OCR) engine is not in the scope of this study, a multilayer perceptron (MLP) based OCR has been implemented for experimental purposes.

There are other statistical and feature based OCR approaches, however, MLP is selected both for simplicity and performance reasons.

Java 2 Standard Edition 1.4.2 is used as the development platform for experiments and implementation of the proposed system. Although Java platform provides extensive support for loading different image file types, it has very little support for image processing. For that reason, an open-source library called Java Imaging and Graphic Library (JIGL) is utilized for basic image processing operations like convolution and filtering. This library is mostly used for experimental purposes and the final LPR system is built from scratch in order to provide an optimized real-time application.

Experiments are conducted on a personal laptop computer with an Intel® Pentium® 4 CPU at 1.80 GHz and 1 GB of RAM. The implemented LPR system has been tested with images taken from different locations, like highway entrance, parking lot, streets, etc. Using still images instead of streaming video simplified the image acquisition process and enabled the author to focus more on the actual LPR processes. Test images database contain 90 images, each presenting different illumination conditions, plate size and orientation. Images in the size of 768x288, 640x312, 640x480 and 400x300 are used in color and grayscale formats.

In this thesis, I have designed and implemented a real-time LPR system by using Turkish license plate format and geometry features. Block-level edge processing and filtering of irrelevant edges at an early stage have reduced the processing time significantly. Multiple license plate properties like uniform background intensity, aspect ratio, edge density and connected edge blocks are used for candidate evaluation instead of a single property like background color or border around the license plate. This resulted in high accuracy on license plate localization.  In character segmentation and recognition, the license plate format is utilized for resolving conflicts and multiple alternatives.

## 1.4 Outline

This study explored potential directions for implementing a real-time license plate recognition system. In the first chapter, problem definition and environment is explained in detail. The second chapter provides the background information, including the history of developments and previous studies. The third chapter includes experiments, results and the proposed solution. The last chapter summarizes the thesis study with conclusions. Further references and directions are listed at the end of this document.

# CHAPTER 2

# BACKGROUND INFORMATION

Images containing text data provide useful information for annotating, indexing and segmentation of their contents. As a result, Text Information Extraction has been studied for many years and numerous techniques and algorithms have been proposed to address this problem and related issues [21, 25, 26]. Depending on the application domain, text extraction can be extremely challenging due to illumination, background, texture and font size, style, color and alignment.

The text extraction process involves localization, enhancement and recognition of text blocks in a given image. License Plate Recognition systems can be recognized as an application of Text Information Extraction (TIE) algorithms. As the application is specific to license plate images, LPR text extraction process can leverage particular visual features and constraints. Different methods have been proposed on LPR text extraction and each uses a specific set of perceptible features to emphasize the text areas. The following properties are utilized frequently in text extraction processes:

- Geometry: Depending on the application, it is possible to make assumptions on geometrical properties like size, alignment and inter-character distance. Although font size and style can vary a lot in a document, consecutive text blocks tend to have the same size and style properties. In most cases, characters appear in horizontally aligned clusters, however, camera position or object alignment can cause perspective distortions.

- Color and Texture: Characters in a text block tend to have the same color and texture properties. Connected-component and wavelet analysis methods make it possible to detect and localize text blocks using this feature.

- Motion: In video, the caption texts remain in the view and consecutive frames contain same texts in the same position. This information can be utilized in tracking and enhancement of text blocks.

- Edge: Background and foreground colors of text blocks in an image should provide sufficient contrast in order the characters to be readable. This results in strong edges at the text boundaries.

## 2.1 Text Localization

Textual information in video frames can be classified contain into two types; caption texts and scene texts. Caption texts can be defined as the artificial text blocks overlaid on the image, whereas, scene texts are natural parts of the image. Example caption text and scene images are depicted in Figure 4. In LPR systems, the focus is located on scene text extraction, since the license plate text lies in the image naturally.

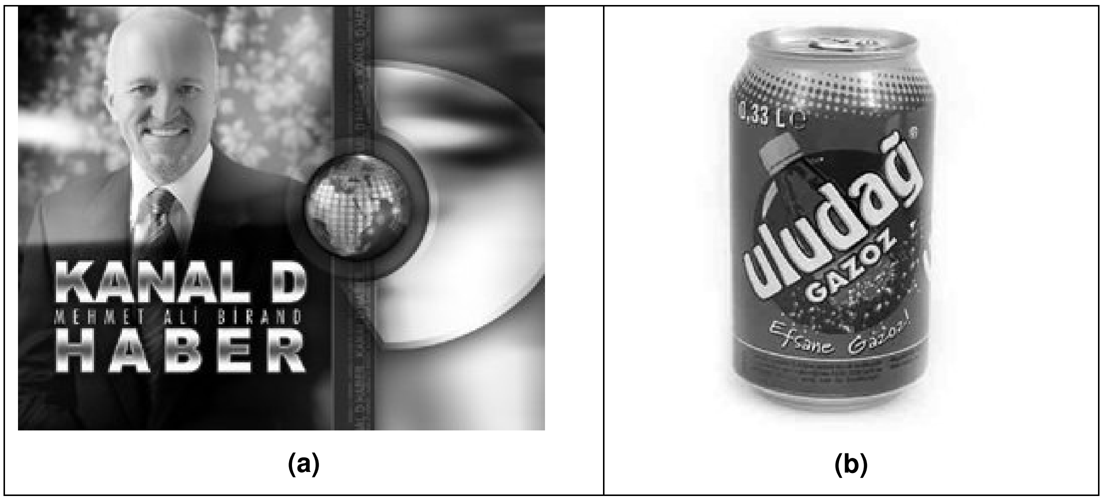


**Figure 4 Caption and scene text examples**
**a) Caption text, (b) scene text**

Localization of scene text is more difficult than caption text localization since the frame difference does not provide any hints, and lighting conditions as well as perspective distortions reduce the accuracy of text localization. Scene texts can have any orientation, hence, it becomes more difficult to determine whether a given image block contains textual or some other information. Zhong et al. [27] proposed a connected component based (CC-based) method applied after color reduction. Jain and Yu [28] used another CC-based method after multi-valued color image decomposition. Hasan and Karam [29] proposed a morphological approach, whereas Li et al. [30] utilized texture information for scene text localization.

These proposed methods can be grouped into categories.  Each category will be explained in the following sections.

### 2.1.1 Connected Component Based Methods

Connected component based methods use a bottom-up approach by grouping pixels and components into larger components until all the regions are identified. Geometrical analysis and spatial arrangement features are used to filter the non-text components and mark the text regions. Although each CC-based method utilizes different approaches, there are four common steps:

i. preprocessing, such as color clustering and noise reduction
ii. connected component generation
iii. filtering non-textual components
iv. component grouping

Zhong [27] used a CC-based method which uses color reduction. This method assumes that text regions occupy a significant portion of an image. Shim et al. [31] utilized intensity homogeneity of text regions and grouped the pixels with similar intensity values. After removing large objects that are marked as backgrounds, each candidate region is subjected to verification using size, area, fill factor and contrast. Lienhart et al. [32] regarded text regions as connected components with the same or similar color and size, and applied motion analysis to enhance the results. Segments, which are too small or too large, are filtered out.

Due to their relatively simple implementation, CC-based methods are widely used in text information extraction. On the other hand, CC-based methods can segment a character into multiple components in case of polychrome texts and low-resolution noisy images. Moreover, several input-dependent threshold values are required to filter out non-text components on the image.

### 2.1.2 Edge Based Methods

Edge-based methods focus on high contrast between the text and the background. The edges of the text boundary are identified and merged, and then several heuristics are used to filter out the non-text regions. Usually, an edge filter (e.g., Sobel, Canny) is used for the edge detection, and a smoothing operation or a morphological operator is used for the merging stage.

Smith and Kanade[33] apply a 3x3 horizontal differential filter and perform thresholding to locate the vertical edges. After eliminating the small edges, adjacent edges are connected and a bounding box is computed. Afterwards, heuristics like aspect ratio and size are applied to each candidate in order to filter non-text block. Sato, Kanade et al. [25] combine closed caption extraction with super-imposed caption (artificial text) extraction. The text extraction algorithm is based on the fact that text consists of strokes with high contrast. It searches for vertical edges which are grouped into rectangles. The authors recognized the necessity to improve the quality of the text before passing an OCR step.

### 2.1.3 Texture Based Methods

Texture-based methods use the distinct textural properties that distinguish the text blocks from the background. The techniques based on Gabor filters, Wavelet, FFT, spatial variance, etc. can be used to detect the textural properties of a text region in an image.

Wu et al. [34] segment an input image using a multi-scale texture segmentation scheme. Potential text regions are detected based on nine second-order Gaussian derivatives. A non-linear transformation is applied to each filtered image. The local energy estimates, computed at each pixel using the output of the nonlinear transformation, are then clustered using the K-means algorithm. Mao et al. [35] propose a texture-based text localization method using Wavelet transform. Harr Wavelet decomposition is used to define local energy variations in the image.

The problem with traditional texture-based methods is their computational complexity in the texture classification stage, which accounts for most of the processing time. In particular, texture-based filtering methods require an exhaustive scan of the input image to detect and localize text regions. This makes the convolution operation computationally expensive.

## 2.2 Related Studies

There have been many studies in this field and different methods have been proposed. The majority of the research is done on plate localization, which is very specific to the LPR domain.

Lee at al. [2] located the moving objects like motorcycles by removing the stationary image blocks. Afterwards, the low contrast blocks are filtered and a projection based method is used for plate localization. The main weakness of this method is the sensitivity to non-text edges on moving objects. Another problem is that the method is designed for moving vehicles, stationery vehicles and their plates will not be detected.

Duan et al. [3] combined the contour algorithm with the Hough transform for plate localization. The computational cost of Hough transform is reduced by applying the transform only to contours; however, there can be many contours to be processed. This method was not suitable for image without high contrast at license plate borders.

Yang et al. [5] used a fixed background color to find the possible license plate regions. Similar colored areas are filtered using edge density filters and remaining areas are processed for recognition. This method requires high quality color images taken in daylight, thus, dark images, changing ambient light and background color as well as similar vehicle background result in very poor localization rates.

Yu and Kim [6] used vertical edge features for plate localization. The algorithm uses the assumption that license plates have a relatively clear outline. Hence, algorithm tries to extract long vertical edges surrounding the plate region. However, problems arise when this outline is not very clear, which is the case in most Turkish license plate images.

Sirithinaphong et al. [7] applied a 4-layer back-propagation neural network on the input image after binarization. A supervised learning stage is conducted with certified license plates and vehicles. The method is highly dependent on image quality, camera position and the aspect ratio of the license plate. The neural network needs to be trained after registering new cars.

Emiris and Koulouriotis [8] proposed a license plate recognition method in semi-structured environments. This method is based on the fact that license plate information resides in a known region and small rotations are permitted.

Rahman et al. [9] combined color constraints and edge feature for plate localization. This method is capable of correcting small tilts and operates in real-time. The prototype implementation assumed a clear license plate text and further tests need to be conducted on this system.

Hontani and Koga [11] proposed another plate localization method based on ridge-type points. It is assumed that the characters in the license plate contain ridge-like elements and such areas are marked as text candidates.

Ghao and Zhou [14] enhanced the image before processing and geometrical features are used for localization. A priori knowledge of proper character size is used in filtering the non-character regions. Candidate regions are combined and a confidence value is calculated. The region with the highest confidence value is selected as the plate region.

Poon et al. [18] used a morphology based approach for plate localization. This method provides poor results under low illumination. Kim et al. [19] utilized a learning based method based on color and texture feature. Nijhuis et al. [20] also utilized a similar approach, but also combined fuzzy logic in their implementation. This method is fast and accurate for many images, but problems arise when the same color is encountered in the background. Moreover, such strict color constrains impede license plates with different colors to be detected by the system.

Kim et al. [35] proposed a genetic algorithm based segmentation method using HLS (Hue-Lightness-Saturation) transformed color values in the gene representation. This method involves a many iterations and consumes much time in the segmentation step. Brugge et al. [4] used a Discrete-Time Cellular Neural Network (DTCNN) for plate segmentation. This method used morphological operators and different structuring elements in segmentation process, requiring an expensive post processing. Many other methods [10-12] and algorithms [15-17, 19] have been proposed using similar features and approaches.

# CHAPTER 3

# LICENSE PLATE RECOGNITION SYSTEM

License plate recognition starts with the acquisition of images from an image source, namely from a surveillance camera. In this thesis, this stage is omitted for simplicity and still images are used instead. The operation continues with the localization of the license plate candidate regions. Afterwards, the resulting regions are enhanced and rotation correction is applied when necessary. Character segmentation operation is performed on the enhanced and corrected image blocks to extract individual characters, which are sent to the recognition engine for identification. Finally, the characters are identified and verified against the given license plate formats. In the following sections, each operation and related experiments are presented.

## 3.1 License Plate Extraction

Localization of the license plate area is the most difficult and time-consuming operation in the LPR system. In order an LPR system to operate in real-time, this stage must be really fast and accurate.

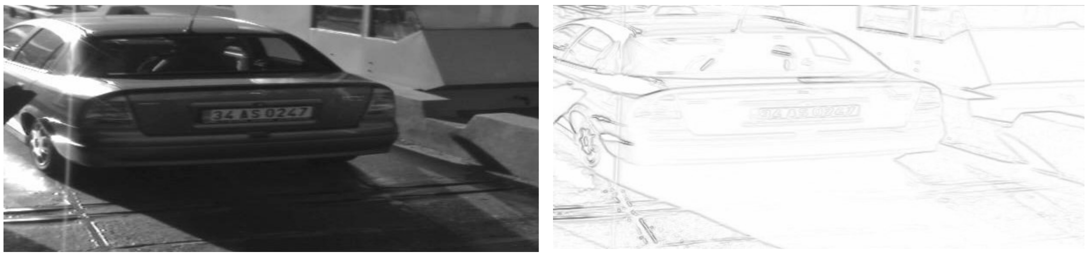
The proposed localization method is based on vertical edge features and involves pre-processing of the image for enhancement. The collected edge features are transformed into edge blocks and edge strength values are calculated for each block. These blocks are filtered by a threshold and further elimination is done by applying constraints to each of them. The noisy regions are separated from text blocks by analyzing the connected edge pixels and high similarity in the background intensity.

### 3.1.1 Image Enhancement

Various problems and different challenges are encountered in license plate images. Although grey-scale images are used in many LPR applications, some experiments also include color images. In this thesis, color images are converted to gray-scale before processing using the following formula.

$$Y = 0.299\,R + 0.587\,G + 0.114\,B$$

The acquired images need processing and enhancement before the localization operation. Since the proposed method is based on vertical edge features, the image needs to be sharpened in order to emphasize the high contrast between the license plate text and the background. Without such improvements, the license plate region does not provide enough edge information for detection as shown in Figure 5.



**(a)** **(b)**

**Figure 5 Sample image and its edge map**
**(a) acquired image, (b) Sobel edges without preprocessing**

Several sharpening algorithms have been used in the experiments and each produced undesired side-effects on sample images. In the first experiment, a widely used 3x3 unsharp mask is applied to the acquired image. The resulting image and its vertical Sobel edges are shown in Figure 6.

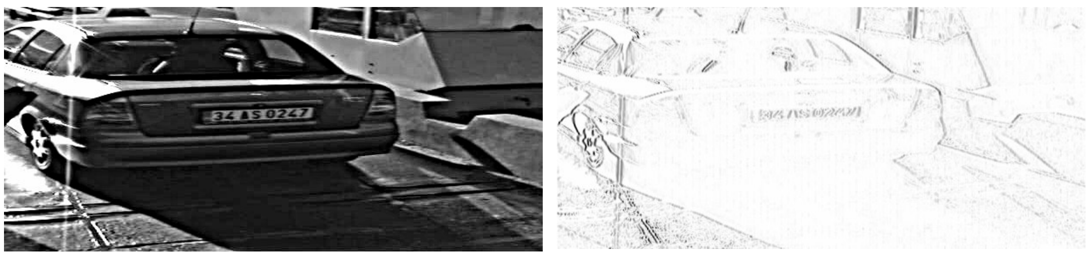


**Figure 6 Sharpened image and its edge map**
**(a) sharpened with unsharp mask, (b) Vertical Sobel edges after processing**

The text edges become clearer on the new edge image. Although the enhancement is remarkable and the edge result is quite satisfying, this method is not very suitable for a real-time LPR system, since the edge residues introduced by the method cause computationally complex filtering overheads.

In the second experiment, histogram equalization method is used to enhance the image. However, in some images, the plate text becomes unreadable after histogram equalization as illustrated in Figure 7. Contrast stretching method produced a very similar output on the same image.

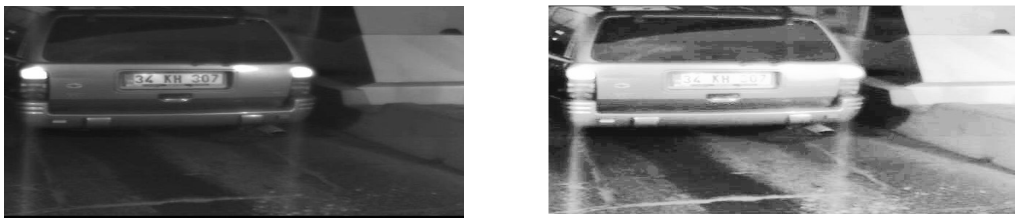


**Figure 7 Histogram equalization**
**(a) original image, (b) histogram equalized image,**

It is believed that the failure of these methods is due to the color distribution in the images. When most of the pixels reside on one portion of the histogram, these image enhancement methods fail terribly.

The main focus in the proposed text localization algorithm is based-on the vertical edges in an image. Hence, the main purpose is to enhance the license plate area as much as possible without enhancing other parts of the image. To achieve this, the contrast difference must be calculated for each region and only the regions with higher contrast average than the selected threshold are enhanced with the contrast stretching method. Several experiments with this method are performed with different windows sizes and the resulting images are shown in the Fig 8.

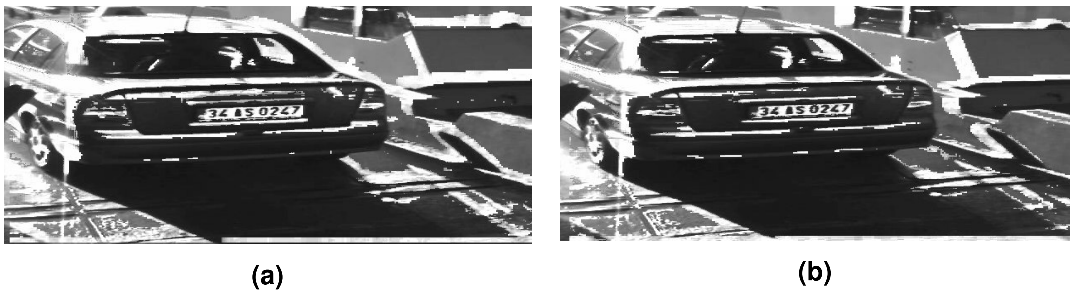
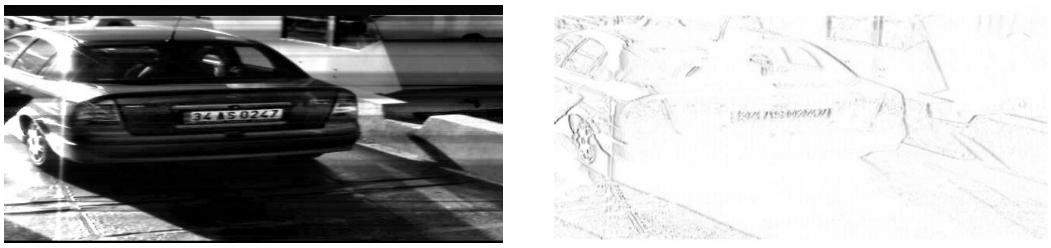


**Figure 8 Contrast streching**
**(a) 16x16 region contrast stretching, (b) 24x8 region contrast stretching**

In this method, unlike others, successive pixels with the same color in the original image can have different color values in the enhanced image. As a result, with small region sizes, this method produces artificial inter-block edges in the resulting image. Artificial edges are not actually a part of the original image. They are generated as a side effect of local image enhancement algorithm.

These generated edges can introduce a lot of false positives for the text localization engine, hence consumes a lot of computational power upon edge filtering. When overlapping regions are used (like convolution), the execution time increases enormously and renders the system inoperable in real-time. Another approach is to apply edge detection directly to each region after enhancement. This method also provided very good results, however, blocks with weak edges may become very strong after the enhancement. As a result, more false positives are encountered.

As the window height decreases and the window width increases, the algorithm provides smoother results. It is observed that the proposed method substantially enhances the plates with relatively low contrasts. After processing the example image above, the enhanced image and resulting edge map are illustrated in Figure 9.



(a) (b)

**Figure 9 Modified contrast streching**
**(a) {width}x1 contrast stretching, (b) vertical Sobel edges after processing**

With the maximum width (image width) and the minimum height (1 pixel) values are utilized in the algorithm, a smooth resulting image is obtained. Since there is only one region on a given row, there exist no strong inter-block edges produced by the smaller windows as shown in Figure 8.

After comparing the accumulated experiment results, the last method is selected for image enhancement. This method performs as much enhancement as the unsharp mask operation, yet it does not introduce as much irrelevant edge remnants. In addition, further tests with the other sample images produced similar desired outputs.

#### 3.1.1.1 Modified Contrast Stretching

Contrast stretching (also called Normalization) attempts to improve an image by stretching the range of intensity values to make full use of possible values. Unlike histogram equalization, contrast stretching is restricted to a linear mapping of input and output values.

The first step is to determine the limits over which image intensity values will be extended. The histogram of the original image is examined to determine the value limits. The lower and upper bounds in the unmodified picture are stored in *l* and *h* respectively. If the original range covers the full possible set of values, straightforward contrast stretching will achieve nothing, but even then sometimes most of the image data is contained within a restricted range; this restricted range can be stretched linearly, with original values which lie outside the range being set to the appropriate limit of the extended output range. Let the target intensity range be *(a, b)*, then for each pixel, the original value *p* is mapped to output value *o* using the function:

$$C(o) = \frac{(p-l)(b-a)}{h-l} + a$$

In the modified contrast stretching method, *l* and *h* values are extended by a factor of $\sigma \in [0, 1)$, which denotes the percentage of histogram data to be cropped from each side (left and right). Let $\omega$ be the number of pixels in the given region. The number of pixels to be cropped is calculated as $\tau = \omega * \sigma$. The *l* value is incremented until the number of pixels in the cropped histogram exceeds $\tau$. The same operation is done for the *h* value, but decremented in each iteration. With this method, more sharpening is achieved and clearer results are obtained.

### 3.1.2 License Plate Localization

The vertical edge features are extracted by applying the vertical Sobel operator (convolution mask) shown in Figure 10.

$$G(y) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 2 \end{bmatrix}$$

**Figure 10 The vertical Sobel convolution mask**

The resulting edge map is filtered out using a selected threshold value. From the priori knowledge, it is known that the license plate text must provide a high contrast from the background. In the implementation, this threshold value is selected to be 40. The resulting image is shown in Figure 11.

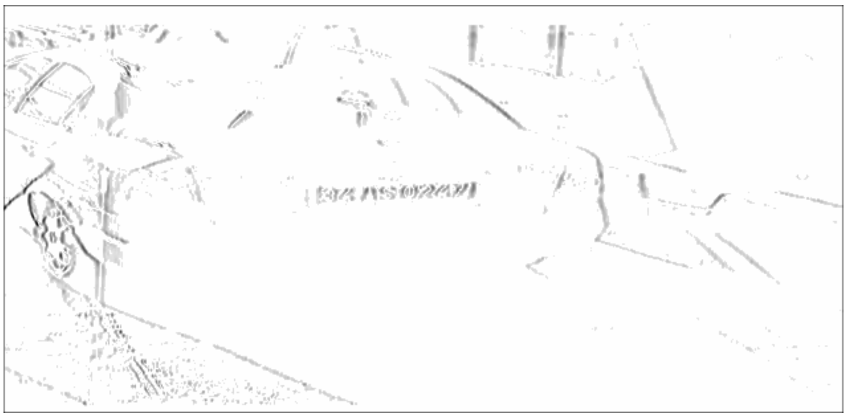


**Figure 11 The filtered edge map of the example image**

Instead of working in pixel-level, the edge image is divided into blocks for faster processing. Different block sizes have been used in the experiments. As the block sizes increase, the accuracy of the text boundaries decreases accordingly. Hence, the optimal block height is the minimal possible value, which is 1.

The filtered edge map is divided into 12x3 windows for further processing. For each block, the number of edge pixels is counted and their strength values are accumulated. The calculated edge blocks are shown in Figure12. Blocks with edge pixels less than the threshold value are filtered out. Then, an average edge strength value is calculated for every block. Connected blocks with similar edge strengths are merged and each segmented component is marked with combined edge strength.

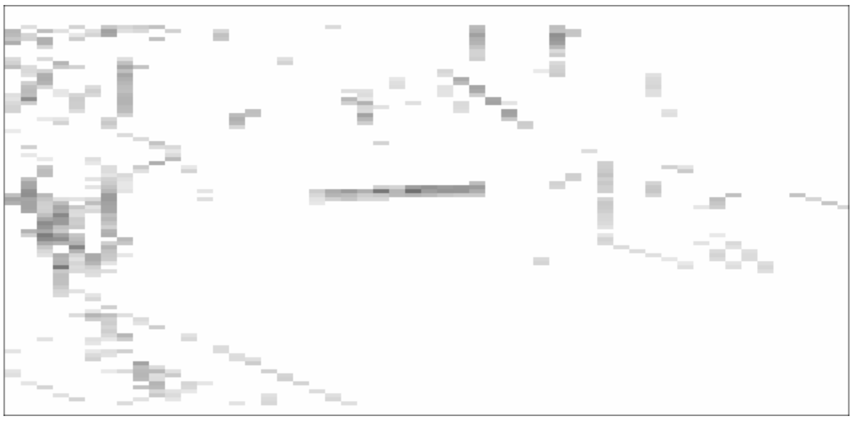


**Figure 12 The edge blocks after thresholding**

Although vertical projection of the edge blocks seems to be a good candidate for locating the plate text in this example, this method fails terribly on other images. Instead, a heuristics based split-and-merge algorithm is used for localizing the plate region. The priori knowledge entered in the LPR system is used in the heuristics. The values given in the priori knowledge are calculated by estimating the minimum and maximum value ranges each property can take with the given camera positioning. The property details are shown in Table 3.1.

After applying the heuristic and constraints, blocks with no neighbors and noisy blocks are removed. Neighboring blocks with similar edge strengths are merged. A rank value, based on how well each connected edge component fits into the given heuristics, is calculated for each component. The component with the highest rank is selected as the license plate candidate region as shown in the Figure 13.

**Table 3.1 Priori knowledge properties**

| **Property** | **Min** | **Max** | **Description** |
|---|---|---|---|
| Plate Width | 70px | 400px | Plate width is estimated to be somewhere in this range. |
| Plate Height | 10px | 40px | Plate height is estimated to be somewhere in this range. |
| Max Y Shift | 0px | 10px | Maximum Y shift tolerated between two connected plate blocks. |
| Max Space/Height | 0 | 2.5 | The space/height ratio determine the maximum horizontal space between two connected plate blocks |
| Width / Height | 6 | 12 | The expected width/height aspect ratio |
| Plate Formats | - | - | Valid plate character formats. These formats are utilized in character recognition and validation. |

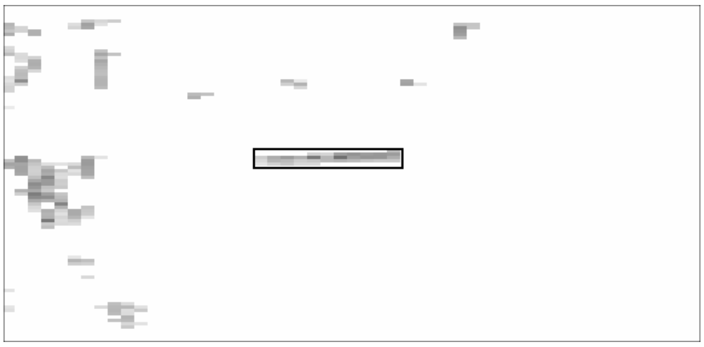


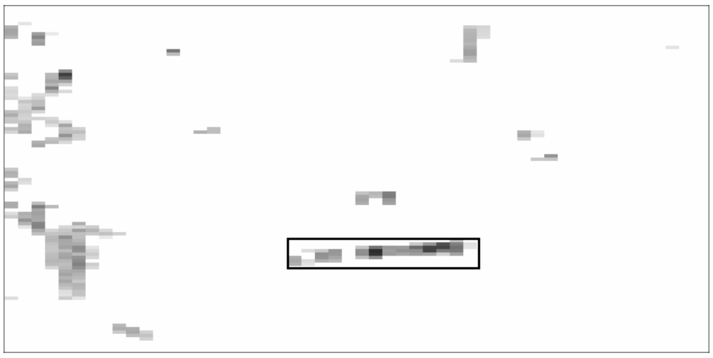
**Figure 13 Plate candidate region with the highest rank**

The following properties are used in rank calculation.

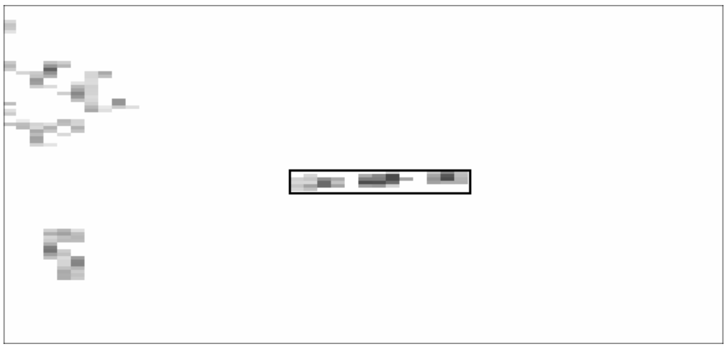
i. Average edge density of individual blocks in a component
ii. Aspect ratio
iii. Connected edge pixel density

iv. Nearly uniform background
v. Block position (The nearer to the image center, the better)

In most of the cases, the highest ranked candidate region contains the license plate. In the cases otherwise, the remaining candidate areas can be processed in ranking order. Other examples are shown in Figure 14.



**(a)**



**(b)**

**Figure 14 Examples for block merges**
**(a) two blocks merged, (b) three blocks merged**

## 3.2 Character Segmentation

The license plate region is roughly located in the previous step and further processing is required before character segmentation. The plate image must be corrected in case of a rotation due to camera positioning or perspective distortion. Afterwards, the corrected plate image is analyzed in detail and surrounding non-plate regions are cropped. Intensity normalization and sharpening operations are applied to the remaining plate image in order to improve the accuracy of character segmentation. The plate region shown in Figure 14a will be used as an example in the following sub-sections. The original image and extracted plate region is shown in Figure 15.

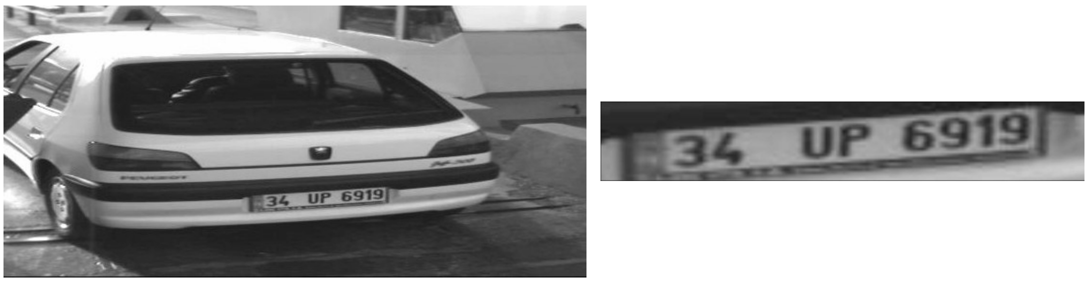


**Figure 15 Extracted license plate image**
**(a) the original image, (b) extracted plate region (zoomed)**

### 3.2.1 License Plate Correction

Unlike caption texts, which are artificially overlaid onto images in one direction (horizontally in most of the cases), license plate characters can be found in different orientation in different images. For that reason, the rotation amount needs to determined and corrected before the segmentation process.

#### 3.2.1.1 Hough Transform

The Hough transform is a technique for detecting lines or curves in a picture by applying a coordinate transformation, such that all the points belonging to a curve of a given type map into a single location in the transformed space [Rosenfeld & Kak, 1982]. The transformed space is quantized into an accumulator array for representation, and for each pixel of interest, votes are added to the array locations associated to the transformed coordinates matching each hypothetical curve as shown in Figure 16.

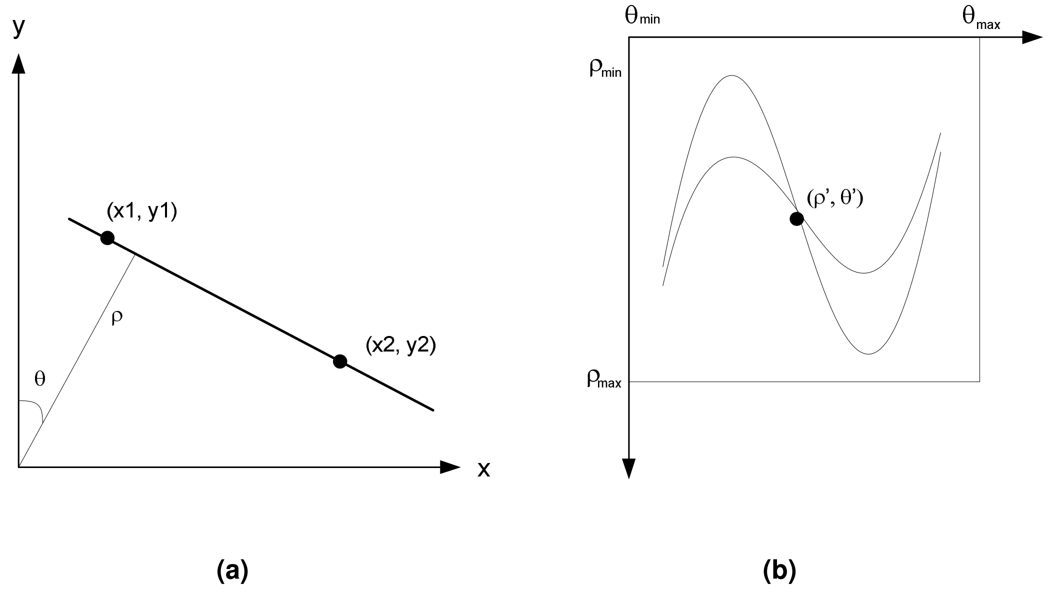


**Figure 16 a) Normal representation of a line and (b) accumulator array in ρ-θ parameter space**

Consider a foreground pixel, in position *(x1, y1)* in Figure 16. All possible lines passing through this point must satisfy the equation $y1 = mx1 + c$. We may regard *(x1, y1)* as fixed, so the equation corresponds to a line in *m-c* parameter space. In the case of lines parallel to the y-axis, m would be infinite, so it is convenient to use the parametric equation for a line: $\rho = x1 \cos\theta + y1 \sin\theta$. By regarding *(x1, y1)* as fixed, the equation describes a sinusoidal curve in *(ρ, θ)* space. Now consider a second point *(x2, y2)* that will also have an associated sinusoidal curve in *(ρ, θ)* space. The curves will intersect at a point *(ρ', θ')*, which represents a line in *(x, y)* space connecting the two points. The Hough technique starts by initializing a *(ρ, θ)* accumulator array, with fixed increments for both parameters. Each point in *(x, y)* space is analyzed and all the (ρ, θ) locations satisfying the equation $\rho = x \cos\theta + y \sin\theta$ are incremented by one. The local maxima in the accumulator array correspond to collinear points in the image array [36]. The

Hough transform technique may be generalized to fit curves with more than two parameters, although this results in a multi-dimensional accumulator array, which exponentially increases computational cost.

The Hough transform is applied to the example plate region and the θ value with highest hit in the accumulator array is expected to reflect the amount of rotation in the image. However, it is observed that different θ values have similar hit counts. Limiting the θ values in the ranges close horizontal and vertical axis does not reduce the candidates. Calculating the sum of neighboring cells at each θ value improves the accuracy, however, the problem still remains for some test images. In addition, this process requires a lot of computations even after caching the *sin* and *cos* values for each possible θ.

Instead of working in pixel-level, it is possible to use the individual edge blocks in the plate region for calculating the amount of rotation. Applying Hough transform to edge blocks produced very good results.

#### 3.2.1.2 Weighted Regression

Changing the processing from pixel-level to edge blocks simplified the operation, so that regression methods can be utilized for the rotation detection. By using the weighted robust regression method, a precise value can be calculated in *O(n)* complexity where *n* is the number of edge blocks in the plate region.

Linear regression is a method of estimating the conditional expected value of one variable y given the values of some other variable or variables x. The term 'Linear' refers to the assumption of a linear relationship between y and x, which could be modeled $y = \alpha + \beta x$. In the linear regression model, this formula is extended to:

$$y = \alpha + \beta x + \varepsilon$$

The right hand side may take other forms, but generally comprises a linear combination of the parameters, here denoted α and β. The term ε represents the unpredicted or unexplained variation in the dependent variable (y). It is conventionally called the "error" whether it is really a measurement error or not. The error term is conventionally assumed to have expected value equal to zero, as a nonzero expected value could be absorbed into α.

Robust regression an alternative regression method, in which mean absolute error is minimized instead of mean squared error as in linear regression. The calculations are done using the following formulas [37].

$$S_x = x_1 + x_2 + \ldots + x_n$$

$$S_{xx} = x_1^2 + x_2^2 + \ldots + x_n^2$$

$$S_{xy} = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n$$

$$\beta = \frac{nS_{xy} - S_x S_y}{nS_{xx} - S_x S_x}$$

$$\alpha = \frac{S_y - \beta S_x}{n}$$

By reorganizing the formula for ε,

$$\varepsilon_i = y_i - \alpha - \beta x_i$$

The rotation amount is calculated by the $\theta = a\tan(\phi)$, where $\phi$ is calculates as:

$$\phi = \frac{S_{xy} - \frac{S_x S_y}{n}}{S_{xx} - \frac{S_x S_x}{n}}$$

When the divider is zero, $\phi$ is set to $\frac{\pi}{2}$.

After calculating the rotation amount, the plate region is corrected reversing the rotation with the calculated value. The resulting plate image is shown in Figure 17.

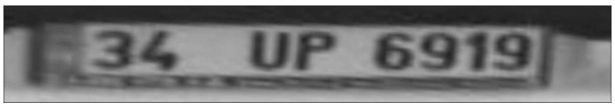


**Figure 17 Corrected plate image**

### 3.2.2 Character Segmentation

Character segmentation is the process of decomposing an image containing a sequence of characters into sub-images of individual symbols. In LPR systems, character segmentation is used to identify the bounding rectangles of each character in the plate image.

The character segmentation methods proposed in previous TIE studies can be applied to the same problem in LPR domain. In the previous studies, three different strategies are employed:

i. Splitting the image into meaningful components
ii. Recognition based character segmentation
iii. Holistic methods for word segmentation (recognition based)

The latter two methods involves recognition for segmentation, hence conflicts with the initial objective. As a result, the first strategy is the only appropriate strategy for the proposed LPR system. In order to achieve a precise segmentation result, the plate image is scaled by a factor of 4, creating an image four times larger than the corrected image. Afterwards, contrast stretching with $\sigma = 0.2$ is applied.

#### 3.2.2.1 Vertical Projection

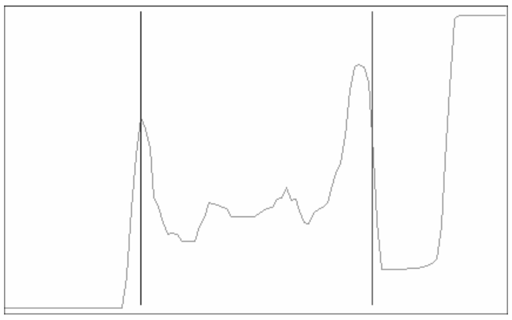
Before performing the character segmentation, the corrected plate image should be trimmed by removing the non-plate regions located at the top and the bottom. The vertical projection of a text is a simple running count of the white pixels in each row (assume that the text is rendered on a white background). For the reverse-colored license plates, this projection will produce limited peaks in the mid-parts of the histogram. As a result, another projection is calculated with the background color black and the plate is marked as reverse-colored. The vertical projection of an image is calculated as follows.

Let $I(x, y)$ be the image where $0 < x < width$ and $0 < y < height$. For every row, the value of $F(y)$ is calculated.

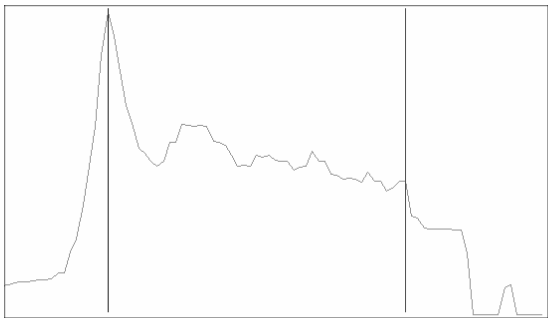
$$F(y) = \sum_{y=0}^{height-1} T(I(x, y))$$

$$T(\rho) = \begin{bmatrix} 0 \ \ when\ \rho\ is\ black \\ 1 \ \ when\ \rho\ is\ white \end{bmatrix}$$

The vertical projection of the example plate is illustrated in Figure 18a.



**(a)**



**(b)**

**Figure 18 (a) The vertical projection of the example plate image, (b) another example**

In many plate images, the vertical projection peaks are obvious, however, more complicated histograms can be encountered in low quality plate images and such cases need to be handled. An example is shown in Figure 18b. The implemented algorithm is tested with different samples and proven to be successful in such difficult cases. The resulting image is shown in Figure 19. The resulting image is negated for reverse-colored plates that are marked before trimming.

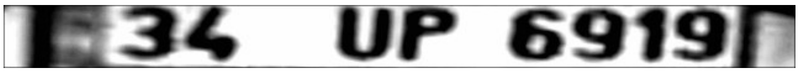


**Figure 19 The example plate image after trimming**

The filtering operation is performed by analyzing the histogram from left-to-right for lower bound and right-to-left for the upper bound. The image is split into three parts. The mid-part is assumed to be a part of the license plate unless no valid filtering point is found by the algorithm and other parts are processed for vertical plate boundaries. The algorithm runs for twice for each part and it is pretty straight-forward.

Pseudo code for the trimming operation:

```
refValue = -1
peak = -1
For each value in the given part of the histogram
      value = histogram[current index]
      If (value > refValue)
            peak = current index
      else if (value < max / 6)
            reset peak values (refValue = -1, peak = -1)
      else if (value < previous value)
            refValue += max / 30
return peak
```

### 3.2.2.2 Horizontal Projection

Space delimiters are used to separate the consecutive characters in license plates. Although the size of the inter-character space can vary from one plate to another, it provides enough information for segmentation in most of the cases. In addition, license plate characters tent to have the same height. On the other hand, the character width can vary depending on the shape of the symbol. For example, the character W can occupy more space than the character ‘I’.

The horizontal projection (similar to "vertical projection") of a text, consists of a simple running count of the black pixels in each column (assume that the text is rendered with black pixels on a white background). This information can be used for detection of white space between successive letters. The columns with minimal or no black pixels can be marked as spaces. When characters touch or overlap horizontally, the projection often contains a minimum at the proper segmentation column. A threshold value is used for separating such characters. On the other hand, a high threshold can result in problems with characters like ‘J’, ‘T’, and ‘L’, which contain only a few pixels on the horizontal direction.

The analysis of a horizontal projection has been utilized as a basis for character segmentation. The horizontal histogram of the example plate is shown in Figure 20.

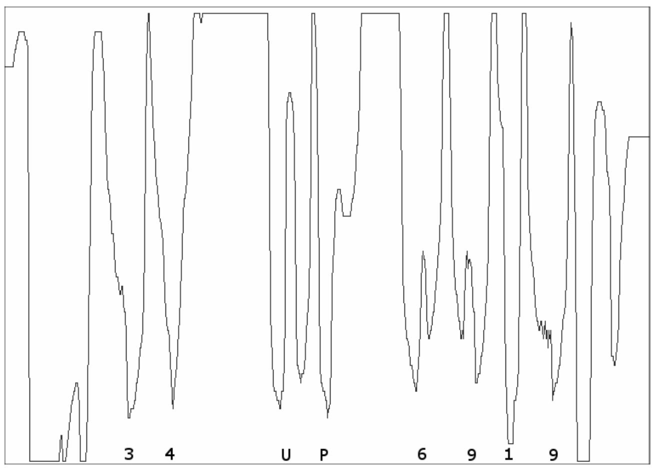


**Figure 20 The horizontal projection of the example plate image**

Character segmentation process requires a two-dimensional analysis. Non-touching characters may not be separable along a single straight line due to overlapping characters. A common approach is based on determining connected black regions ("connected components"). Further processing may then be necessary to combine or split these components into character images. There are two different approaches, one based on the bounding box and the other based on detailed analysis of the connected components.

The distribution of bounding boxes tells a great deal about the proper segmentation of an image consisting of non-cursive characters. Adjacency relationships, size and aspect ratio is used to perform merging and splitting operations. Much of the segmentation task can be accurately performed at a low cost in computation. This approach is very well-suited for LPR systems.

The connected component approach is better for applications requiring segmentation of cursive characters. In license plates images, such character fonts are never used (or rarely encountered).

Let $I(x, y)$ be the image where $x \in [0, width)$ and $y \in [0, height)$. For every column, the value of $F(x)$ is calculated.

$$F(x) = \sum_{x=0}^{width-1} T(I(x, y))$$

$$T(\rho) = \begin{bmatrix} \mathit{0} & when\ \rho\ is\ black \\ \mathit{1} & when\ \rho\ is\ white \end{bmatrix}$$

The character segmentation algorithm works as follows. First of all, it calculates an average character width by analyzing the distance between consecutive peak values. Values higher than the 95% of the average peak value are considered to be in the peak area. Although this threshold value compensates the noise and low quality plate images, it can also produce undesired splits as shown in Figure 21.

Long spaces between characters provide great hints for recognition and verification. The location of spaces can determine the possible plate formats before the recognition. By using this information, the recognition algorithm can resolve character conflicts and prioritize the identification order.

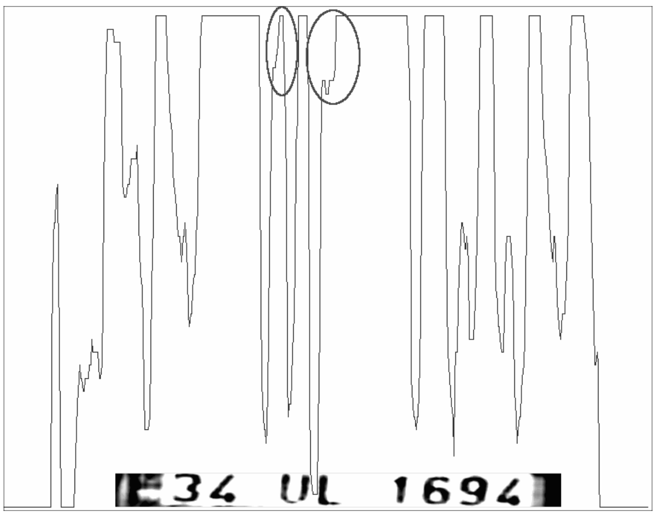


**Figure 21 Undesired split example**

## 3.3 Character Recognition

Initially, the character recognition implementation was not in the thesis plan; instead, an external recognition engine would be utilized for identifying the segmented characters. An open source Optical Character Recognition (OCR) engine called GOCR [jocr.sourceforge.net] was used in the beginning. The initial results were not promising.

The author decided to enhance the image and clear the character as much as possible before sending it to the OCR engine. The results were improved a little, yet still not good enough for identification. A typical Turkish license plate contains minimum 7 characters, however, GOCR were able recognize 3 characters at maximum. Afterwards, a commercial OCR called OmniPage was used for identification. Again, the results were not sufficient for license plate identification. Furthermore, launching an external program for identifying each character consumes a considerable amount of time.

Implementing a robust and efficient OCR engine is a totally different research subject and requires a lot of expertise. In this study, a very limited OCR functionality is implemented in order to complete the LPR application. A neural network based approach is used in the OCR as it is utilized by many open-source and commercial OCRs [39, 40] as well as LPR systems [16]. Moreover, NN-based OCR implementations respond to identification queries in real-time. With the built-in OCR support, the time consumption of the overall system could be calculated precisely.

### 3.3.1 The Perceptron

The perceptron is a kind of binary classifier that maps its input $x$ (a vector of type Real) to an output value $f(x)$ (a scalar of type Real) calculated as;

$$f(x) = \langle w, x \rangle + b$$

where w is a vector of real-valued weights and $\langle \cdot, \cdot \rangle$ denotes dot product. (The dot product is used for computing a weighted sum). $b$ is the 'bias', a constant term that does not depend on any input value.

The sign of $f(x)$ is used to classify $x$ as either a positive or a negative instance, in the case of a binary classification problem. The bias can be thought of as offsetting the activation function, or giving the output neuron a "base" level of activity. If b is negative, then the weighted combination of inputs must produce a positive value greater than $-b$ in order to push the classifier neuron over the 0 threshold. Spatially, the bias alters the position (though not the orientation) of the decision boundary.

Since the inputs are fed directly to the output unit via the weighted connections, the perceptron can be considered the simplest kind of feed-forward neural network.

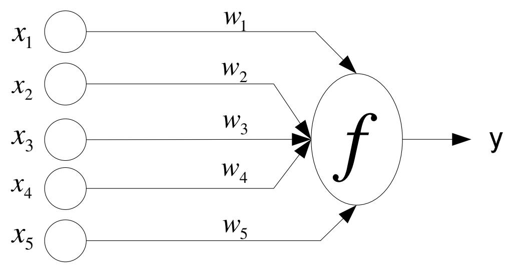


**Figure 22 A Perceptron**

### 3.3.2 Multi-Layer Perceptron Networks

Multi-Layer Perceptron (MLP) is generally used to describe any feed-forward (no recurrent connections) network. The nonlinear model is very general and can, in principle, represent almost anything. There is the trade-off between representational power and efficiency, raising the practical problem of finding a suitably restricted subset of functions $f$ which would have good representational capacity but would also form a space with a structure regular enough to enable efficient learning in practice.

The MLP network is composed of neurons which are very close to the ones represented in the case of the linear network. The linear neurons are modified so that a slight nonlinearity is added after the linear summation. The output $y$ of each neuron is thus;

$$y = \phi\left(\sum_i w_i x_i + b\right)$$

where $x_i$ are the inputs of the neuron and $w_i$ are the weights of the neuron. The nonlinear function $\phi$ is called the activation function as it determines the activation level of the neuron. This refers to interpreting the activation as the pulse rate of biological neurons.

Due to the nonlinear activation function, a multi-layer network is not equivalent to any one-layer structure with the same activation function. In fact, it has been shown that one layer of suitable nonlinear neurons followed by a linear layer can approximate any nonlinear function with arbitrary accuracy, given enough nonlinear neurons.

The activation functions most widely used are the hyperbolic tangent $\tan(h)$ and logistic sigmoid $1/(1+e^{-x})$. These activation functions are used for their convenient mathematical properties and because they have a roughly linear behavior around origin, which means that it is easy to represent close-to-linear mappings with the MLP network.

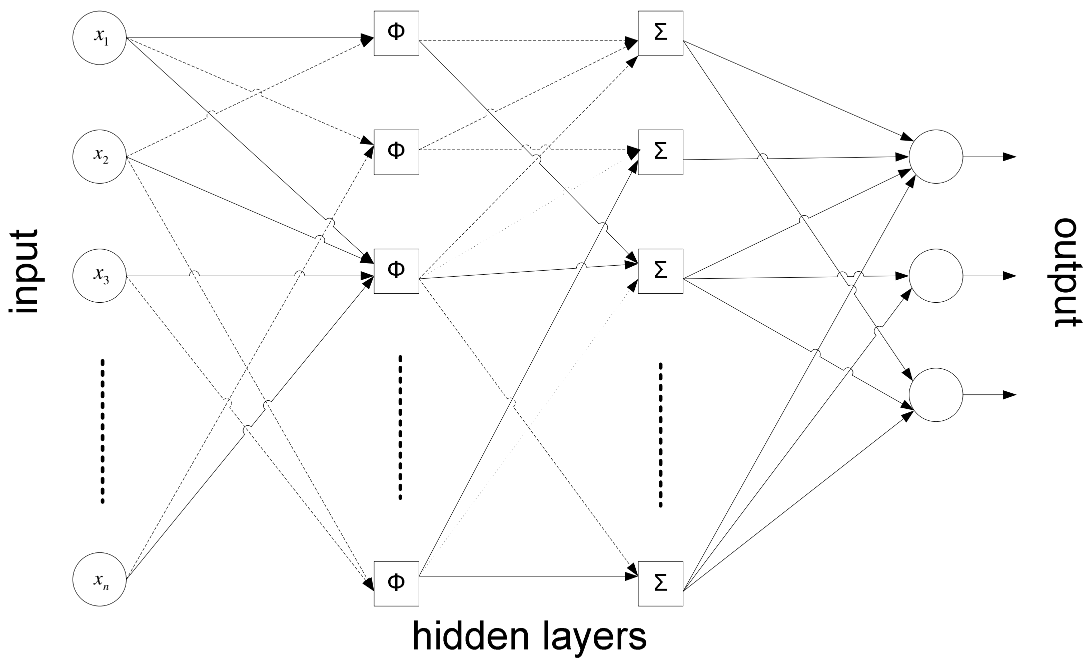


**Figure 23 An example MLP Network**

### 3.3.3 MLP Neural Network Training

The MLP network should be trained with a set of sample images before using it for character recognition. Neural networks brings the possibility of *learning*, which in practice means the following:

Let $F$ be the class of functions and $f \in F$ be the optimal function which solves the given recognition task. The cost function $C : F \rightarrow \Re$ is a measure of how far away we are from an optimal solution to the problem that we want to solve. Learning algorithms search through the solution space in order to find a function that has the smallest possible cost.

$$C = \frac{1}{N} \sum_{i=1}^{N} \left| f(x_i) - y_i \right|^2$$

$N$ is the number of samples used in the training and $y_i$ is the expected output for the given input. When $N$ is large enough and the number of nodes in the neural network is sufficient, the cost can be reduced to a certain level. At each iteration, a delta rule for the change to the weight $\Delta w_{ij}$ from node *i* to *j* is calculated and applied.

$\Delta w_{ij} = \eta * \delta_j * y_i$ where $\eta$ denotes the learning rate and the local gradient $\delta_j$ is defined as follows:

$\delta_j = \phi(x_j) \sum_k \delta_k w_{kj}$ where *k* ranges over those nodes for which $w_{kj}$ is non-zero (i.e. nodes *k* that actually have connections from node *j*. The $\delta_k$ values have already been computed as they are in the output layer (or a layer closer to the output layer than node *j*)

In the developed neural network, $\eta$ is selected as 0.9 for fast learning. Smaller $\eta$ values results is slower learning but they accurately follow the path of steepest descent in weight space. When the $\eta$ value is large, the algorithm may oscilate and can not converger to a suitable local minima. The sigmoid function $1/(1+e^{-x})$ is selected as the $\phi$ function in our implementation.

As the number of training samples increases, it becomes harder for the neural network to align its weights in a short period of time. In order to overcome this, a distinct neural network is trained for each possible character in the license plate

(A-Z, 0-9). These neural networks are trained in advance and saved to disk for later recognition.

The number of input nodes is dependent on the number of pixels in the sample images. That is, each input node is connected to a single pixel value in the image. Smaller images (less input nodes) results in faster learning and convergence when the same number hidden nodes is utilized. As the number of input nodes increases, the required nodes in the hidden layer should also be increased in order to achieve the same leaning rate.

The input image size also determines how much information will be processed in the neural network. The larger and cleaner the image is, the better is the recognition. On the other hand, when the segmented character images are smaller than the expected input size, scaling them to a larger size has no positive affect on the recognition. After several iterations, the image size 9x12 is selected as the input size. This is the minimal images size where the scaled down character images can be easily distinguished by the eye. The number of hidden nodes is selected to be 27 (number of input nodes / 4).

The training sample characters are normalized to 9x12 and then binarized using a threshold ($\tau = 128$). Resulting images are fed into every neural network with an expected value 1 if the neural network is created for the given character and 0 otherwise.

### 3.3.4 Recognition

After character segmentation, individual characters were extracted from the license plate. Each character image is normalized to 9x12 and then binarized using a threshold ($\tau = 128$). The resulting image is fed into each distinct neural network and all the resulting values greater than a certain threshold (0.9) is accumulated into an array. In many cases, only one neural network returns a value greater than the threshold. In such cases, the character is recognized directly.

When more than one neural network calculates to a value grater than the threshold, the license plate format is used for prioritization. With the help of this format information, it becomes possible to differentiate the following characters at recognition time: ‘B’ and ‘8’, ‘G’ and ‘6’, ‘I’ and ‘1’, ‘Z’ and ‘2’.

Example recognized character images are show in Figure 24.

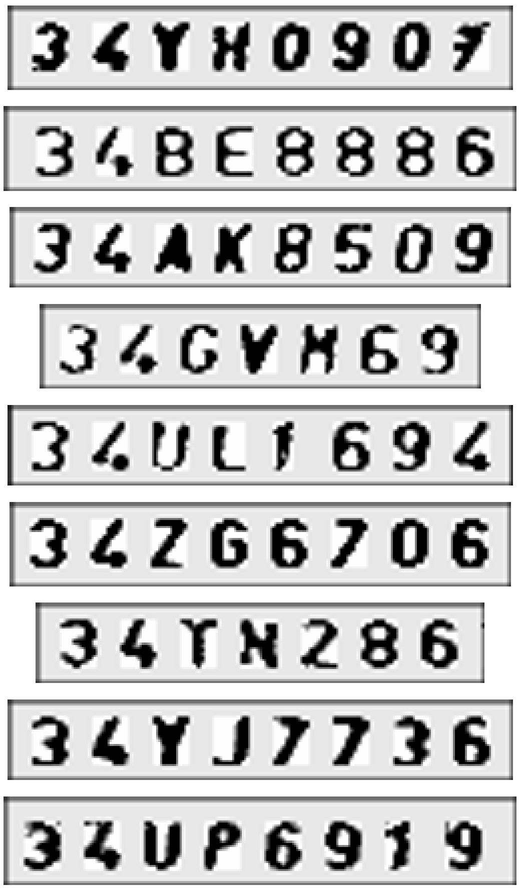
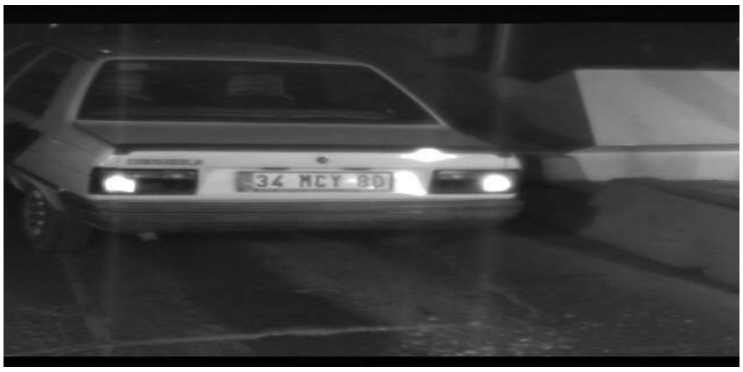


**Figure 24 Recognized Character Image Examples**

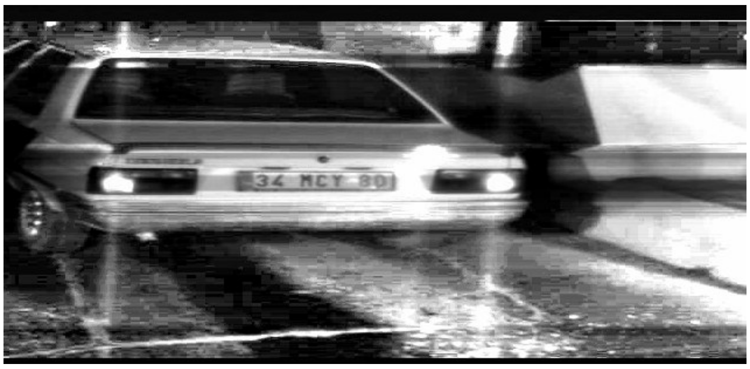
## 3.4 Unrecognized License Plates

Although the implemented algorithm worked quite well for most of the test images, it was unable to locate the license plate properly for the image shown in Figure

25a. Image enhancement algorithm was unable to improve the image due to poor illumination conditions. The enhanced image is shown in 25b.



**a) Original Image**



**b) Enhanced image**

**Figure 25 Unrecognized License Plate Examples**

The plate location algorithm found the license plate partially at the proper location, but some parts of the plate image is cropped as a result of poor vertical edges. The partially found license plate is show in the Figure 26.

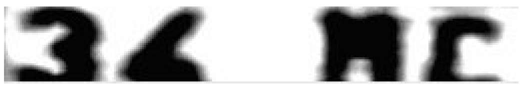


**Figure 26 Partially Located License Plate**

# CHAPTER 4

# CONCLUSION

Due to its wide application domain, License Plate Recognition systems have received a lot of attention from both the academic and commercial worlds. Unlike strictly constrained predecessors, I have implemented a working prototype with soft constraints like license plate geometry and format. This system is capable of processing images in real-time and the average processing time was calculated to be around 200 milliseconds on an up-to-date notebook.

In this thesis, I have explored potential techniques and algorithms for implementing a real-time LPR. Although there are different methods for achieving similar results, the algorithm selection plays an important role on execution time. As a result, several image enhancement, correction and segmentation methods were utilized during experiments. Modified and adapted versions of selected methods have been implemented in the prototype.

With proper constrains, LPR systems can identify vehicles very fast and produce accurate results. In this study, Turkish license plate format and constrains were used for optimization purposes. License plates from other countries were also used for testing and it was observed that the system was also able to locate foreign license plates. The character recognition rate was measured to be lower on foreign license plates since they contain different fonts and character sizes.

A number of strategies have been introduced to reduce the time complexity of the proposed LPR algorithm. Initial enhancement operations reduced the processing time by ignoring irrelevant edges at an early stage. Discontinued and small edges were filtered before the license plate localization procedure. On the other hand, filtered blocks may partially contain license plate areas. This problem was solved by expanding the license plate area after localization.

In the future, additional constraints can be built and applied for faster and more precise identification. In my opinion, the major problem for LPR systems still resides on the character recognition. Due to poor image quality and high variance in illumination conditions, this task remains a highly challenging subject in the LPR domain.

# REFERENCES

1. Shyang-Lih Chang, Li-Shien Chen, Yun-Chung Chung, and Sei-Wan Chen, Automatic License Plate Recognition, IEEE Transactions on Intelligent Transportation Systems, 2004, Vol. 5, Nr. 1.

2. Hsi-Jian Lee, Si-Yuan Chen and Shen-Zheng Wang, Extraction and Recognition of License Plates of Motorcycles and Vehicles on Highways, Proc. of the 17th International Conference on Pattern Recognition, 2004.

3. Tran Duc Duan, Tran Le Hong Du, Tran Vinh Phuoc and Nguyen Viet Hoang, Building an Automatic Vehicle License-Plate Recognition System, Proc. of the Intl. Conf. in Computer Science RIVF, 2005, pp. 59-65.

4. M. H. ter Brugge, J. H. Stevens, J.A. G. Nijhuis and L. Spaanenburg, License Plate Recognition Using DTCNNs, Proc. of the Fifth IEEE International Workshop on Cellular Neural Networks and their Applications, 1998, pp. 212-217.

5. Yao-Quan Yang, Jie Bai, Rui-Li Tian and Na Liu, A Vehicle License Plate Recognition System Based on Fixed Color Collocation, Proc. of the Fourth International Conference on Machine Learning and Cybernetics, 2005, pp. 5394-5397.

6. Mei Yu and Yong Deak Kim, An Approach to Korean License Plate Recognition Based on Vertical Edge Matching, Proc. of the IEEE International Conference on Systems, Man, and Cybernetics, vol. 4, pp. 2975-2980.

7. Thanongsak Sirithinaphong and Kosin Chamnongthai, The Recognition of Car License Plate for Automatic Parking System, Proc. of the Fifth International Symposium on Signal Processing and its Applications, 1999, pp. 455-457.

8. D.M. Emiris and D.E. Koulouriotis, Automated Optic Recognition of Alphanumeric Content in Car License Plates in a Semi-structured Environment, Proc. of International Conference On Image Processing, 2001, Vol. 3 pp. 50-53.

9. Choudhury A. Rahman, Wael Badawy and Ahmad Radmanesh, A Real Time Vehicle's License Plate Recognition System, Proc. of the IEEE Conference on Advanced Video and Signal Based Surveillance, 2003, pp. 163-166.

10. Takashi Naito, Toshihiko Tsukada, and Kei-ichi Yamada, Moving-Vehicle License Plate Recognition Method Robust to Changes in Lighting Conditions, Systems and Computers in Japan, 2000, Vol.31 pp. 82-91.

11. Hidekata Hontani and Toshio Koga, Character Extraction Method without Prior Knowledge on Size and Position Information, Proc. of the IEEE International Vehicle Electronics Conference, 2001, pp. 67-72.

12. Yuntao Cui and Qian Huang, Character Extraction of License Plates from Video, Proceedings of the Conference on Computer Vision and Pattern Recognition, 1997, pp. 502-507.

13. N. H. C. Yung, K. H. Au and A. H. S. Lai, Recognition of Vehicle Registration Mark on Moving Vehicles in an Outdoor Environment, International Conference on Intelligent Transportation Systems, 1999, pp. 418-422.

14. Da-shan Gao and Jie Zhou, Car License Plates Detection from Complex Scene, Proc. 5th Int. Conf. on Signal Processing, Vol. 2, pp. 1409–1414.

15. Muhammad Sarfraz, Mohammed Jameel Ahmed, and Syed A. Ghazi, Saudi Arabian License Plate Recognition System, Proceedings of the International Conference on Geometric Modeling and Graphics, 2003, pp. 36-41.

16. Hi-Tech Solutions, License Plate Recognition Demo, http://www.htsol.com/Products/Demo.html, Last accessed date: January 18, 2006.

17. Luis Salgado, Jose M. Menendez, Enrique Rendon and Narciso Garcia, Automatic Car Plate Detection and Recognition through Intelligent Vision Engineering, Proceedings. IEEE 33rd Annual 1999 International Carnahan Conference, 2001, Vol. 3, pp. 50-53.

18. Joe C.H. Poon, Muffaddal Ghadiali, Gary M.T. Man, Lau Man Sheung, A Robust Vision System for Vehicle License Plate Recognition Using Grey-Scale Morphology, Proc. of the IEEE International Symp. Industrial Electronics, 1995, vol. 1, pp. 394-399.

19. K.K.Kim, K.I.Kim, J.B.Kim, and H. J.Kim, Learning-Based Approach for License Plate Recognition, Proceedings of the IEEE Signal Processing Society Workshop, 2000, Vol. 2, pp. 614-623.

20. J.A.G Nijhuis, M.H. ter Brugee and K.A. Helmholt, Car License Plate Recognition with Neural Networks and Fuzzy Logic, Proceedings of the ICNN, 1995, Vol. 5, pp. 2232-2236.
21. H. K. Kim, Efficient Automatic Text Location Method and Content-Based Indexing and Structuring of Video Database, Journal of Visual Communication and Image Representation 7, 1996, pp. 336-344.

22. M. Sawaki, H. Murase, and N. Hagita, Automatic Acquisition of Context-based Image Templates for Degraded Character Recognition in Scene Images, Proc. of International Conference on Pattern Recognition, 2000, Vol. 4, pp. 15-18.

23. J. Zhou, D. Lopresti, and T. Tasdizen, Finding Text in Color Images, Proc. of SPIE on Document Recognition V, 1998, pp. 130-140.

24. D. S. Kim and S. I. Chien, Automatic car license plate extraction using modified generalized symmetry transform and image warping, Proc. of the IEEE Int. Symp. Industrial Electronics, 2001, Vol. 3, pp. 2022–2027.

25. T. Sato, T. Kanade, E. K. Hughes, and M. A. Smith, Video OCR for Digital News Archive, Proc. of the IEEE Workshop on Content based Access of Image and Video Databases, 1998, pp. 52-60.

26. J. Ohya, A. Shio, and S. Akamatsu, Recognizing Characters in Scene Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1994, pp. 214-224.

27. Yu Zhong, Kalle Karu, and Anil K. Jain, Locating Text In Complex Color Images, Pattern Recognition, 1995, pp. 1523-1535.

28. A. K. Jain, and B. Yu, Automatic Text Location in Images and Video Frames, Pattern Recognition, 1998, pp. 2055-2076.

29. Yassin M. Y. Hasan and Lina J. Karam, Morphological Text Extraction from Images, IEEE Transactions on Image Processing, 2000, pp. 1978-1983.

30. H. Li, D. Doerman, and O. Kia, Automatic Text Detection and Tracking in Digital Video, IEEE Transactions on Image Processing, 2000, pp. 147-156.

31. J. C. Shim, C. Dorai, and R. Bolle, Automatic Text Extraction from Video for Content-based Annotation and Retrieval, Proc. of International Conference on Pattern Recognition, 1998, Vol. 1, pp. 618-620.

32. R. Lienhart and F. Stuber, Automatic Text Recognition In Digital Videos, Proc. of SPIE, 1996, pp. 180-188.

33. M.A. Smith and T. Kanade, Video Skimming for Quick Browsing Based on Audio and Image Characterization, Technical Report, 1995.

34. V. Wu, R. Manmatha, and E. M. Riseman, An Automatic System to Detect and Recognize Text in Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999, pp. 1224-1229.

35. S. K. Kim, D. W. Kim, and H. J. Kim, A recognition of vehicle license plate using a genetic algorithm based segmentation, Proc. Int. Conf. Image Processing, 1996, Vol. 2, pp. 661–664.

36. D. Ballard and C. Brown, Computer Vision. Englewood Cliffs, N J: Prentice-Hall, 1982.

37. M. Kustermann, License Plates of the World, http://www.worldlicenseplates.com, Last accessed date: Feb 4, 2006,

38. X. F. Hermida, F. M. Rodriguez, J. L. F. Lijo, F. P. Sande, and M. P. Iglesias, “A system for the automatic and real time recognition of VLP’s (Vehicle License Plate),” in Lecture Notes in Computer Science, 1997, Vol. 1311, pp. 552–558.

39. Abby FineReader, Abby OCR / ICR / OMR, data capture and Linguistic Software, http://www.abbyy.com, Last accessed date: May 8, 2006.

40. GOCR, GOCR Home, http://jocr.sourceforge.net, Last accessed date: May 8, 2006.

41. JavaOCR, JavaOCR Home, http:// www.javaocr.com, Last accessed date: May 16, 2006.

42. Nuance, Nuance - OmmniPage, http://www.nuance.com/omnipage, Last accessed date: June 2, 2006.