ISTANBUL TECHNICAL UNIVERSITY | MATHEMATICAL ENGINEERING DEPARTMENT

MAT 116E ADVANCE SCIENTIFIC AND ENGINEERING COMPUTING PROJECT

PLATE RECOGNİTİON SYSTEM by USING MATLAB

Kaan CAN | 090190311 | CRN: 20662

1. Abstract

Today, motor vehicles play a very important role in transportation. The number of vehicles participating in traffic is increasing day by day and this situation makes traffic management difficult. One of the most valuable tools that facilitate traffic management is vehicle tracking systems. One of the cheapest methods developed for vehicle tracking is image plate recognition systems. Unlike other vehicle tracking systems, license plate recognition systems function with the simple camera and software without the need for expensive detectors, radars, RF tags, and satellite systems. Thanks to the cheapness of the system, more points can be observed in the traffic and it can be a very practical information source for taking actions to increase the flow rate and safety of traffic. With this motivation, this project, it is aimed to develop a simple license plate recognition system application for civil vehicles that comply with Turkish license plate standards by using image processing techniques on MATLAB. The application designed in this project detects the plate in the uploaded picture, reads its content, and displays the result and the information of which province it belongs to the vehicle.

2. Method

The project consists of 4 sub-stages. These determine the plate's position in the image, separating the characters (numbers and letters) on the plate, defining each character by comparing it with the database, and finally adding a graphical user interface. The program performs these tasks in turn and the plate is defined. Various image processing tasks are performed with various MATLAB functions and methods for each stage.

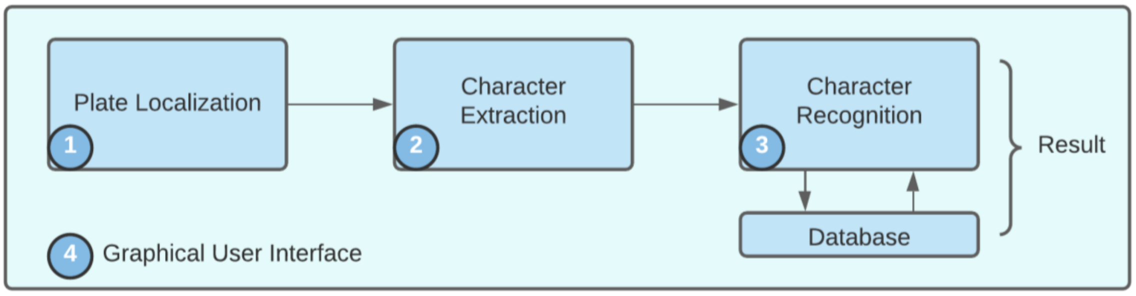


Figure 1: method diagram

2.1. Plate Localization

It aims to detect the position of the vehicle's license plate area in the image. In this way, many unnecessary details in the image are eliminated and a simpler image is obtained for the next stage. The correct determination of the location of the plate area is very important for the efficiency of the program. If wrong cropping is done, useless material will be sent for the next stages and reading cannot be performed correctly.

Various image processing techniques are used to define the plate region. First, the image is resized with the **resize** function. Each pixel value has RGB color values. The **rgb2gray** function converts the image to gray tones by obtaining a two-dimensional matrix by averaging the values of each pixel color component between 0-255. This grayscale image also applies the "Thresholding" algorithm with the **imbinarize** function, bringing each pixel to a black-and-white level with a value of 0 or 1.

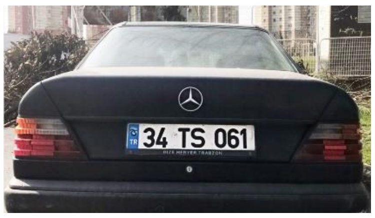


Figure 2: original image

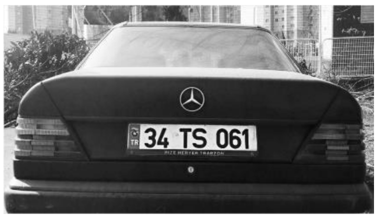


Figure 3: rgb2gray

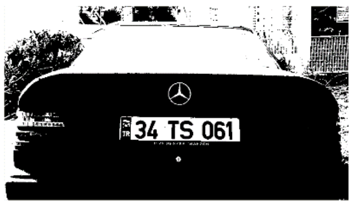


Figure 4: imbinarize

The **edge** function takes the border pixels of the white clusters with the “Sobel edge detection” algorithm and turns the inside of the cluster to black, creating a picture consisting of white borderlines. The **imdilate** and **strel** functions enlarge the white-valued pixels by the desired value, thus correcting the possible breaks in the borderlines.

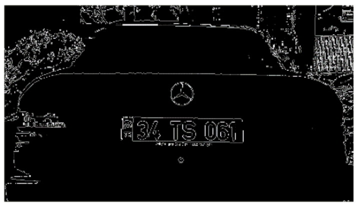


Figure 5: edge

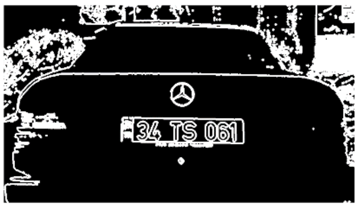


Figure 6: imdilate and strel

With the **imfill** function, the closed-boundary sets are filled with white. The **imerode** function turns the other element around itself according to the dominance of black and white in the image, thus avoiding many unnecessary small white clusters. With all these stages, we get rid of the noise in the image and it remains to find the plate in the processed image that we have left. With the **regionprops** function, the sizes of white clusters are calculated and their positions are recorded. Then, with a simple comparison, the largest white cluster is selected and the positions of the plate are determined. The **imcrop** function crops the boundary values we obtained from the original image, obtains an image with only the plate region, and sends it to the next stage.

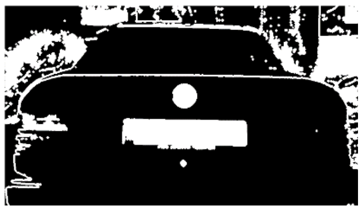


Figure 7: imfill

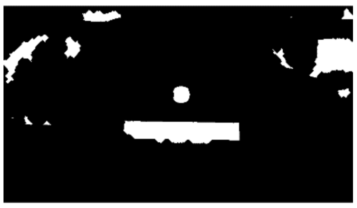


Figure 8: imerode

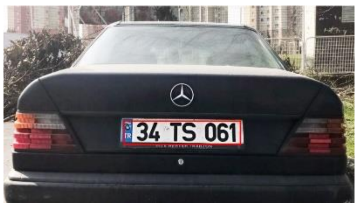


Figure 9: rectangele

2.2. Character Extraction

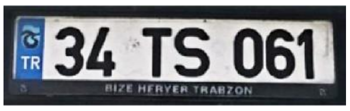
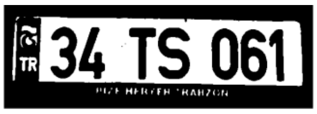
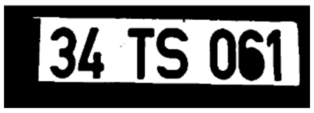


Figure 10: imcrop / source image of second stage

At this stage, it is aimed to find the characters in the plate image taken from the first stage, that is, the separate positions of the numbers and letters, and to distinguish them from each other. First, the noise in the image is extracted, and then the characters are distinguished from each other. At this stage, it is very important to remove and clean unnecessary data other than noise, that is, our characters. If cleaning is not done well enough, many unnecessary and wrong materials may go to the next stage, or on the contrary, the values that should be determined to be characters can be deleted and cause the incomplete transfer.

The plate region image is converted to black and white level with the **imresize, rgb2gray, imbinarize** functions, similar to the previous step. Then, white clusters smaller than the value determined by the **bwareaopen** function are cleaned by converting them to black, in the same way, small black clusters are cleaned by taking the reverse of the picture and all small noises in the image are removed. Afterward, an additional image is created in which the characters are also cleaned with the **bwareaopen** function, and these two images are subtracted from each other to get rid of other big noises and a clean image with only the character values.





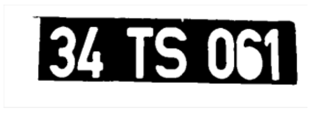


Figure 11: imbinarize **Figure 12:** bwareaopen for White **Figure 13:**bwareaopen for black

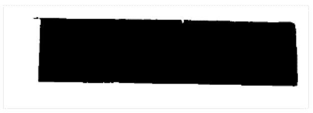
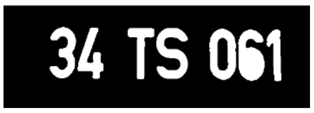



Figure 14: bwareaopen for characters **Figure 15:** subtraction of figures 13 and 14

With the noise removed visual **bwlabel** function, all the white clusters in it are labeled from left to right and the limit values are calculated. With the **regionprops** function, the sizes of character clusters are calculated and their positions are recorded. With the **rectangle** function, the perimeter of each is marked in the image. The location of these characters and the cleaned images are sent to the next stage.

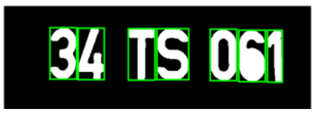


Figure 16: rectangle with regionprops data

2.3. Character Recognition

After the localizationtion and extraction of the characters, all that remains is to understand what they mean and to determine what the characters are. For this, the characters in the image we process are compared with the character image database created with the same method. If a rendered character image is similar to an image in the database with high accuracy, it is concluded that it is the same value. In this comparison system, the more possible examples in the database, the more accurate results will be obtained.

2.3.1. Production of Database Materials

We need a database for comparison, and the Producting_characters.m program has been prepared to create the necessary visual materials for this. This file creates 42x24 visual materials by processing the images we express in a similar way to the second stage. The database is enriched with examples such as the filled versions of characters with spaces such as 0, A, B, and so on.

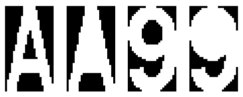


Figure 17: examples of character database materials

2.3.2. Database Creator

Training_imgfildata.m program has been prepared in order to convert the database images we have prepared to be suitable for use in the main program. This program converts the data in the file containing the visual materials into a cell table containing the information about which character each image represents, making it suitable for use in the main program.

2.3.3. Comparison Process

At this stage, the database prepared and the character images from the second stage are compared with each other and aim to read what is written on the plate. For this, firstly, each character whose position is determined is converted into 42x24 dimensions similar to those in the database. Afterward, each image is compared with all the elements in the database with the **corr2** function and the "coleration coefficient" algorithm, and the database element with the highest value, that is, the most similarity, is seen, and its name is taken and this data is recorded. This step is repeated for each character, and the plate is read and interpreted. The value read is examined in the State_founder.m file and information about which vehicle the vehicle belongs to is reached.

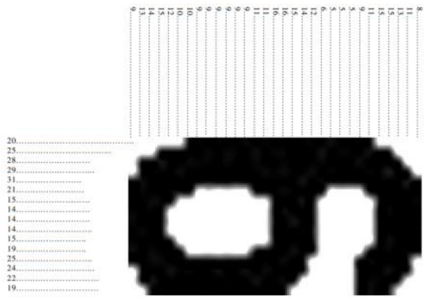


Figure 18: comparison of “6” pixel by pixel with corr2

2.4. Graphical User Interface

At this stage, the target is to turn the program into an application by designing a user-oriented, easy-to-use visual interface. For this, the GUI (Graphical User Interface) toolbox in MATLAB was used.

3. Application

The application designed in this project detects the plate in the uploaded picture, reads its content, and displays the result and the information of which province it belongs to the vehicle.

Guide for how to use Plate Recognition App
1- Open "PlateApp.m" file and run it
2- Press the "Load Image" button
3- Select an image with include a plate and see the results
4- You can repeat 2.,3. steps as much as you want

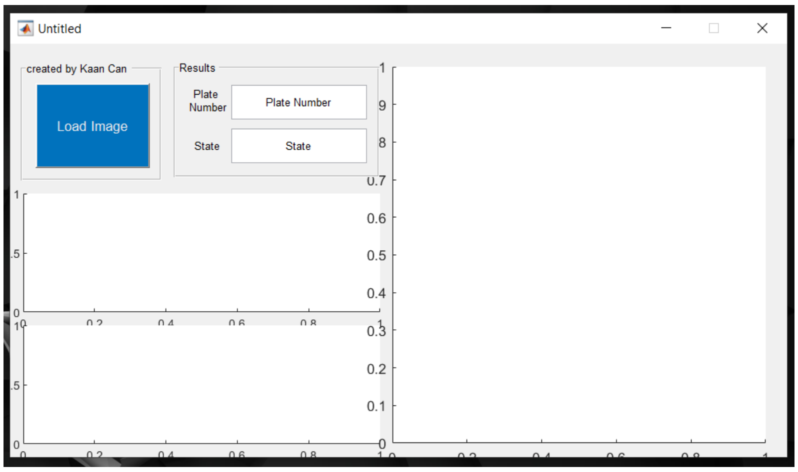
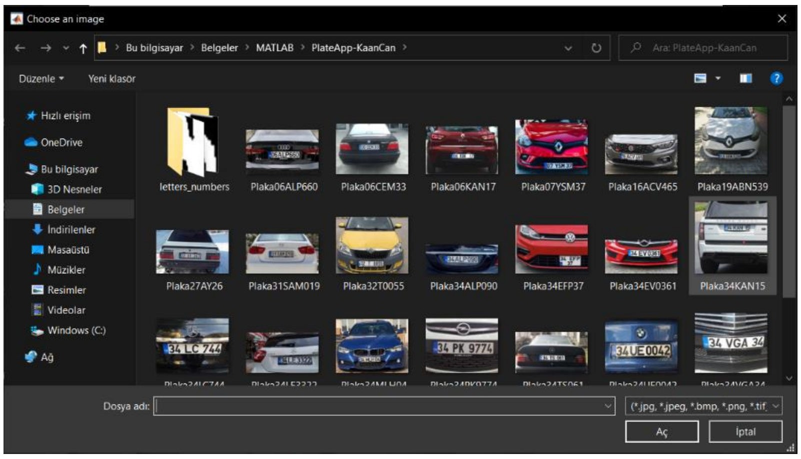


Plate Number: Plate Number
State: State



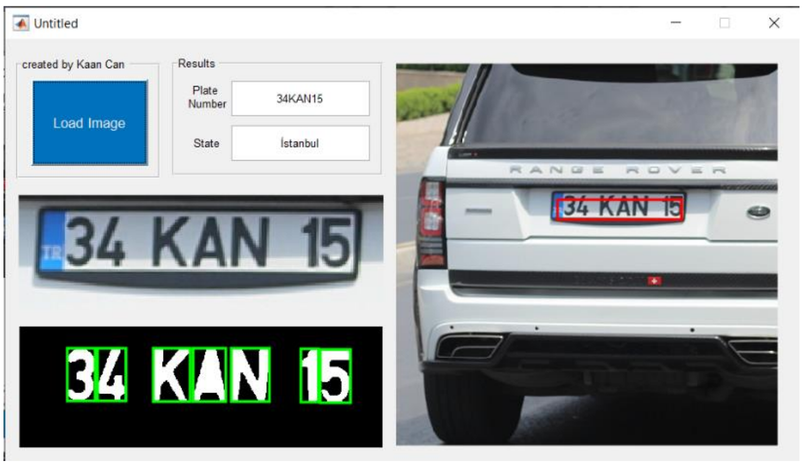


Plate Number: 34KAN15
State: İstanbul

Other examples:

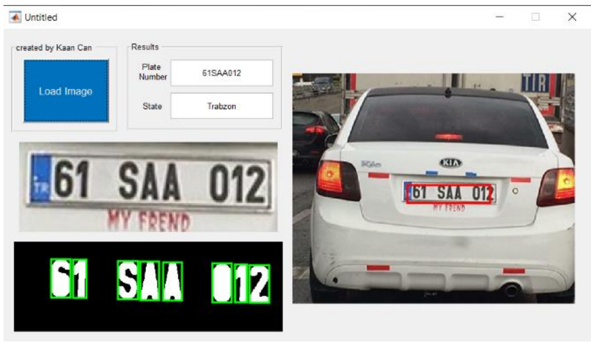


Plate Number 61SAA012

State Trabzon

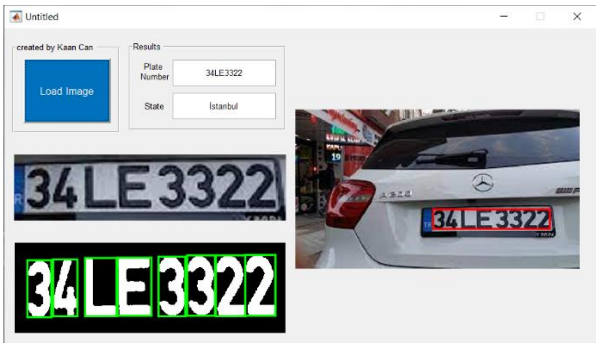


Plate Number 34LE3322

State İstanbul

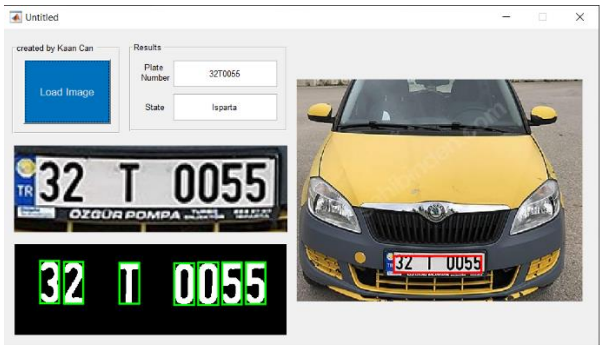


Plate Number 32T0055

State Isparta

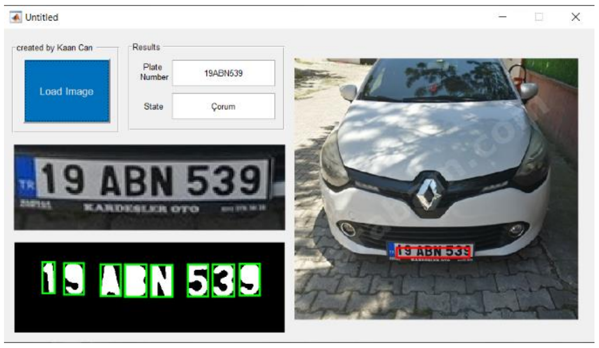


Plate Number 19ABN539

State Çorum

4. Results and Suggestions

The program works quite successfully, identifies the plate in the relevant image, and accurately tells which one it belongs to. However, there are special conditions that may cause the program to produce erroneous results. If the uploaded image has a very low resolution, pollution, shadows, etc., which will affect the legibility of the plate. In the relevant photo, the extremely bright surfaces and the grid structures of some vehicle models may be considered more dominant than the license plate while detecting the plate position, and the plate position may not be detected correctly. Another situation that puts the application in trouble is that the plates drawn from a rather crooked angle cannot be read because they create outputs that do not resemble the database. Additional functions can be produced and the application can be strengthened to remove the troubles that these three special conditions may cause. An application can be developed to work simultaneously with the video camera and to recognize vehicles with official and foreign license plates. Even so, the application can be a useful tool for many projects that need license plate recognition.

5. References

- **“Recognize Text Using Optical Character Recognition (OCR) - MATLAB & Simulink.”** *MathWorks*, 2009,www.mathworks.com/help/vision/ug/recognize-text-using-optical-character-recognition-ocr.html.
- **“Recognize Text Using Optical Character Recognition - MATLAB Ocr.”** *MathWorks*, 2007,www.mathworks.com/help/vision/ref/ocr.html#bt548t1-2_1.B. YALIM1, N. DOĞAN, **“Türk Taşıt Plaka Standartları İçin Plaka Tanıma Sistemi”** *Bilişim Teknolojileri Dergisi*, 1-1, Ankara, January 2008
- K. K. Çevik, A. Çakır, **“Görüntü İşleme Yöntemleriyle Araç Plakalarının Tanınarak Kapı Kontrolünün Gerçekleştirilmesi”** *AKÜ Fen Bilimleri Dergisi*, 31-38, Afyon, January 2011
- S. Gülenç, **“Gerçek Zamanlı Araç Plaka Tanıma Sistemi Tasarımı ve Gerçeklenmesi”** *Bachelor’s Degree Thesis, Uludağ Unv.* Bursa, 2018
- F. G. Yaşar, A. Alaybeyoğlu, **“A Design of Automatic Vehicle Plate, Color and Brand Recognition System”** *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, 11-1, İzmir, 2018
- B. Göçerler, **“Yapay Zeka Tabanlı Plaka Tanıma ile Bariyer Kontrolü”** *Bachelor’s Degree Thesis, Trakya Unv.* Edirne, 2015
- Abdul Ghani. **“Automatic License Plate Recognition System - Matlab (Image Processing Algorithm).”** *YouTube*, uploaded by Abdul Ghani, 3 Jan. 2014, www.youtube.com/watch?v=bnQp7cLk7O0&t=61s.
- Oguzhan Cantas. **“Plaka Tanıma Sistemi Matlab (License Plate Recognition).”** *YouTube*, uploaded by Oguzhan Cantas, 27 May 2018, www.youtube.com/watch?v=VYmUPSR-u8M.
- MATLAB. “How to Create a GUI with GUIDE - MATLAB Tutorial.” *YouTube*, uploaded by MATLAB, 28 Sept. 2017, www.youtube.com/watch?v=Ta1uhGEJFBE.
- Angora Tutor. “License Plate Recognition with MATLAB.” *YouTube*, uploaded by Angora Tutor, 18 July 2020, www.youtube.com/watch?v=AJW9hAw8fVM&t=522s.
- Mr.Jubayer. “Car License Plate Reader Using MATLAB , BUET(EEE`10).” *YouTube*, uploaded by Jubayer Mahmud, 22 June 2013, www.youtube.com/watch?v=4v8hJbD_B2s.
- Kode4U Tech. “Matlab Number Plate Detection.” *YouTube*, uploaded by Kode4u, 4 Nov. 2017, www.youtube.com/watch?v=woB2Fd0gkHM.