Automatic Localization and Recognition of License Plate Characters for Indian Vehicles

Satadal Saha 1, Subhadip Basu 2 and Mita Nasipuri 3

1 Dept. of Electronics and Communication Engg., MCKV Institute of Engg., Howrah, E-mail:satadal.saha@ieee.org,
2 Dept. of Computer Science and Engg., Jadavpur University, Kolkata, E-mail: subhadip@ieee.org,
3 Dept. of Computer Science and Engg., Jadavpur University, Kolkata, E-mail: mitas@ieee.org

Abstract: Automatic License Plate Recognition (ALPR) is a challenging area of research due to its importance to variety of commercial applications. ALPR systems are widely implemented for automatic ticketing of vehicles at car parking area, tracking vehicles during traffic signal violations and related applications with huge saving of human energy and cost. The overall problem may be subdivided into three distinct key modules: (a) localization of license plate from vehicle image, (b) segmentation of the characters within the license plate and (c) recognition of segmented characters within the license plate. The main function of the module (a) is to find out the potential regions within the image that may contain the license plate. The function of module (b) is to isolate the foreground characters from the background within the detected license plate region. And the function of the module (c) is to recognize the segments in terms of known characters or digits. Though modules (b) and (c) employ most of the traditional methods available to the technologists, module (a) i.e. localization of potential license plate region(s) from vehicle images is the most challenging task due to the huge variations in size, shape, color, texture and spatial orientations of license plate regions in such images. In general, objective of any ALPR system is to localize potential license plate region(s) from the vehicle images captured through a road-side camera and interpret the segmented characters present therein using an Optical Character Recognition (OCR) system, to get the license number of the vehicle.

Again, an ALPR system can have two varieties: on-line ALPR system and off-line ALPR system. In an online ALPR system, the localization and interpretation of license plates take place instantaneously from the incoming video frames, enabling real-time tracking of moving vehicles through the surveillance camera. On the other hand, an offline ALPR system captures the vehicle images and stores them in a centralized data server for further processing, i.e. for interpretation of vehicle license plates. The objective of the current work falls under the second category of ALPR system. In this work, real time vehicle images are captured from a road-side surveillance camera automatically throughout day and night. The images are stored in a centralized data server. A never ending process takes the stored images sequentially and interprets the license number of the vehicle. An innovative idea using statistical distribution of the vertical edges is used for localization of license plate, connected component labeling is used for segmentation of the characters and template matching using an innovative matching technique is used for recognition of the characters. The performance of the system is measured at the three levels, i.e., localization level, segmentation level and recognition level and the result seems to be quite satisfactory.

Keywords: Vertical edge, Sobel’s operator, CCL algorithm, Otsu’s binarization, Segmentation, Template matching.

1. Introduction

Various methods and techniques have already been developed during last couple of decades for the purpose of efficient localization of license plate regions from offline vehicular images. In general, most of the works on ALPR systems [1]-[4] use the edge property as features for localizing standardized license plate regions. Some of these works [2], [5], [6] capture the image of a vehicle carefully placed in front of a camera occupying the full view of it and taking a clear image of the license plate. But in an unconstrained outdoor environment there may be huge variations in lighting conditions/ wind speed/ pollution levels/ motion etc. that makes localization of true license plate regions even more difficult.

Moreover in the practical scenario there may be multiple vehicles of different types in a single scene along with partial occlusions of the license plates caused by other vehicles and/ or objects, in which case the above methods do not work. In one of the earlier works [1], Rank filter is used for localization of license plate regions giving unsatisfactory result for skewed license plates. An analysis of Swedish license plate is done in [2] using vertical edge detection followed by binarization. This does not give better result for non-uniformly illuminated plates. During the localization phase the position of the characters is used in [3]. It assumes that no significant edge lies near the license plate and characters are disjoint. In Greece the license plate uses shining plate. The bright white background is used as a characteristic for license plate in [4]. A work on localization of Iranian license plate is done in [5]. In [6], W. Jia used mean shift algorithm for localization of license plate giving satisfactory result for license plates
having color different from the body color. Spanish license plate is recognized in [7] using Sobel edge detection operator. It also uses the aspect ratio and distance of the plate from the center of the image as characteristics. But it is constrained for single line license plates. An exhaustive study of plate recognition is done in [8] for different European countries.

In the developed countries as well as in most of the developing countries the attributes of the license plates are strictly maintained by the community. For example, the size of the plate, color of the plate, font face/size/color of each character, spacing between subsequent characters, the number of lines in the license plate, script etc. are maintained very specifically. However, in India, the license plates of the vehicles are not yet standardized across different states, making the localization and subsequent recognition of license plates extremely difficult. This large diversity in the features set of the license plate of Indian vehicles makes its localization a challenging problem for the research community.

Unfortunately, limited works [9]-[10] have been done on localizing the license plates from Indian vehicles. A significant work on localization of license plate has been reported in [11] using the vertical edge component of the image. In one of our earlier works [12], a color based segmentation scheme is implemented for Indian commercial vehicles. An ANN based technique is reported in [13] with an accuracy of 80%. A more generalized text segmentation based approach is reported in [14] which when applied on license plate localization domain provides an efficiency of 88%. Because of the complexities in localizing the license plates for Indian vehicles some new technique needs to be developed for localizing the license plate in Indian scenario. To address these challenges the objective of the current paper is to present a novel technique for automatic localization of license plate not only to enrich the research output but also to be used in full-fledged commercial environment. As a first step of application of ALPR system in a particular state government organization, a technique for identifying the images in which the vehicles have actually violated the traffic red signal has already been reported in [15]. In the second step of the ALPR system, the proposed technique can be implemented to localize the license plate of the vehicles violated the traffic red signal and to recognize the characters therein, thereby developing a complete ALPR system.

2. Overview

Figure 1 shows the schematic diagram of the complete ALPR system. Image acquisition is the process of acquiring the images from a vehicular road using surveillance cameras. These images are stored in a centralized data server for rest of the work. Image preprocessing employs some of the quality improvement techniques to facilitate the localization of the license plate. In the edge detection module vertical edge map is created using Sobel’s vertical edge operator. The license plate localization module operates on the vertical edge map as created by the previous module. The objective of this module is to find out the concentrated vertical edges and mark them for further processing. The selected regions are binarized in the next module to get the foreground characters against the background scene. In the next module the connected foreground components are segmented from the background using connected component labeling algorithm. The connected components are then recognized with a pre-stored library using template matching technique.

![Figure 1. Block diagram of the proposed method](image)

3. Image Acquisition

The image dataset for the current experiment is collected as a part of a demonstration project on Automated Red Light Violation Detection system for a Government traffic monitoring authority of a major metro city in India. Three surveillance cameras were installed at an important road crossing in Kolkata at a height of around ten meters from the road surface. All the surveillance cameras were synchronized with the traffic signaling system such that the camera captures the still snapshots only when the traffic signal is turned RED. All the cameras were focused on the Stop-Line to capture frontal images of vehicles near the Stop-Line on a RED traffic signal. The complete image dataset comprises of more than 15,000 surveillance still snapshots, captured over several days/night in an unconstrained outdoor environment with varying lighting conditions, pollution levels, wind turbulences and vibrations of the camera. 24-bit color bitmap images were captured through CCD cameras with a frame rate of 25 fps and resolution of 704 × 576 pixels. Not all these still snapshots contain vehicle images with a clear view of license plate regions. For the current work, 500 images have been identified that contain complete license plate regions appearing in different orientations in the image frame.
4. Pre-Processing

As described in previous section, true color still snapshots of resolution 704 × 576 pixels were captured through multiple surveillance cameras over day and night with embedded noise and huge variations in image quality. Following preprocessing techniques are implemented in the current work to address the issues mentioned above.

4.1 Rotation

Normally the front face of the hard mount camera makes some angle with the vertical plane representing the front face of the road. This angle remains fixed as long as the camera remains fixed at the position of its mount. Now, when a vehicle stops at the front face of the road it also makes almost the same skew angle as mentioned above. This makes the license plate of the vehicle tilted with some known angle depending on the camera fixation. To localize the license plate as well as to recognize the characters therein the image should be rotated such that the license plate becomes horizontal. In the present work, the camera angles are measured with respect to the stop-line and the images for a specific camera are rotated through the fixed angle corresponding to the particular camera using the following formula.

\[
\begin{bmatrix}
x' \\
y'
\end{bmatrix} = \begin{bmatrix}
\cos(\theta) & \sin(\theta) \\
-\sin(\theta) & \cos(\theta)
\end{bmatrix} \begin{bmatrix}
x \\
y
\end{bmatrix}
\]

The rotation algorithm uses this standard rotation formula used is coordinate geometry and is written in Equation (1), where, any pixel (x, y) is rotated by an angle \( \theta \) to generate a new set of coordinates \((x', y')\) for the said pixel.

4.2 Gray scale conversion

In any 24-bit color image, each pixel contains the Red (R), Green (G) and Blue (B) color components, each consuming 8 bits of information. From these R, G and B components, 8-bit gray value for each pixel position is calculated using the formula written in Equation (2).

\[
gray(i, j) = 0.59*R(i, j) + 0.30*G(i, j) + 0.11*B(i, j)
\]

where, \((i, j)\) indicates the position of a pixel in the image, and \(gray(i, j) \in (0, 255)\). The rest of the procedure is applicable to grey image only.

4.3 Median filtering

Median filter is a non-linear filter, which replaces the gray value of a pixel by the median of the gray values of its neighbors. In this work, a \(3 \times 3\) convolution mask is used to get eight neighbors of a pixel and their corresponding gray values. This operation removes salt-and-pepper noise from the image.

5. Edge Detection

The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the object. It can be shown that under rather general assumptions for an image formation model, discontinuities in image brightness are likely to correspond to

- discontinuities in depth,
- discontinuities in surface orientation,
- changes in material properties and

In the ideal case, as a result of application of edge detection of an image may lead to a set of connected curves that indicate the boundaries of objects, the boundaries of surface markings as well as curves that correspond to discontinuities in surface orientation. Moreover, edge detection of an image may significantly reduce the amount of data to be processed and may therefore filter out information that may be regarded as less relevant, while preserving the important structural properties of an image.

In this work, the edges created by the characters within the license plate are extracted. Sobel’s edge operator [16] is used for detection of edge gradients. It is seen that when the characters of the license number are written horizontally, the vertical edges of each character appear at regular interval and they have a specific height. The pattern and concentration of the vertical edges also remains in conformity with the pattern of the license number. This appearance of vertical edge pattern is statistically seen to occur within the license plate, nowhere else within the natural scene of the image. In the present work, this phenomenon is explored to find the license plate region within the image.

The formula for getting vertical edge gradient is written in Equation (3), where, \(img\) is the enhanced image over which the edge detection algorithm is operated upon, \(V_{mask}\) is the Sobel’s mask for vertical edge detection as given below in Equation (4) and \(gradV\) is the vertical edge gradient.

\[
gradV(y, x) = \sqrt{\sum_{n=-1}^{1} \sum_{m=-1}^{1} V_{mask}(n, m) \cdot img(y + n, x + m)^2/4}
\]

The formula for getting vertical edge gradient is written in Equation (3), where, \(img\) is the enhanced image over which the edge detection algorithm is operated upon, \(V_{mask}\) is the Sobel’s mask for vertical edge detection as given below in Equation (4) and \(gradV\) is the vertical edge gradient.

\[
V_{mask} = \begin{bmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{bmatrix}
\]

Depending on the value of the \(gradV\) the edge image is binarized using the threshold \(\mu_{gradV}\) (mean edge gradient value) and formed the edge image \(img_{edge}\).

Deciding the threshold value for binarization of edges is a key factor. In the current work, a dataset of numerous license plates only have been generated and the binarization method have been extensively applied over them for different values of threshold as seen from the histogram of the images of the license plate.
6. License Plate Localization

In the present work, a novel approach has been made based on prominent vertical edges computed from vehicle images for localization of significant license plate regions. It may be observed from Figure 2, that the pattern of the vertical edges at the license plate region is very dense and prominent. Also, the vertical run-lengths of edge pixels within the license plate regions are almost equal to the height of the characters therein.

Using the aforesaid attributes, the overall localization algorithm may be subdivided into the following intermediate stages, viz., identification of potential band of rows, primary localization of license plate regions based on statistical distribution of vertical edge pixels, refinement of license plate regions based on prominent vertical edges and finally, localization of license plate bounding box by removing the noise segments. The steps are discussed hereunder in an algorithmic approach.

6.1 Algorithm of License Plate Localization

First Stage:
1. For row=1 to height
   . For col=1 to width
     edgeval[row]=edgeval[row]+ edgeval[row,col]
2. For row=1 to height
   . If edgeval[row]>Tmin
     mean[row]=mean of edge pixels’ x- positions in a row
     variance[row]=variance of edge pixels’ x- positions in a row
3. For row=1 to height
   . Find the set of continuous rows satisfying variance[row]>maximum variance (say, Vmax).
     //This set rows gives the probable bands having license plates.
4. For each band,
   . set top=starting row
   . set bottom= ending row.

The value of Tmin and Vmax are calculated from the generated dataset.

Second Stage:
1. For each band,
   . Calculate minimum and maximum values of mean μi (say, μmin and μmax)
   . Calculate maximum value of vi (say, vmax).
2. For each band,
   . set left=(μmin - vmax)
   . set right=(μmax + vmax).
3. For each band,
   . Draw box having diagonal corners (left, top) and (right, bottom)
     //This box will localize the position of license plate

Upto this point what we get is the statistically obtained potential license plate region whose dimension indicates the maximum area in which the license plate can appear.

Third Stage:
1. For each bounding box in second stage,
   . From left to right
     find first prominent vertical edge having height > predefined minimum height (say, Hmin)
     if found, set new_left=left
     From right to left
     find first prominent vertical edge having height > predefined minimum height (say, Hmin)
     if found, set new_right=right
2. Draw box with left-top and right-bottom corners’ coordinates as (new_left, top) and (new_right, bottom)
3. Once all these new bounding boxes are drawn, the overlapped or very close bounding boxes are merged to get common bounding box. This case actually occurs in case of multi-line character set license plate.

The value of Hmin is actually dependent on the height of the characters in the character set and is obtained by averaging the heights of the vertical edges within the bounding box obtained in second stage. The outcome of this stage is the true license plates and along with them there may be additionally generated boxes indicating false license plates. These noisy boxes are removed in the fourth stage depending on the aspect ratio and the area of the generated boxes.

Fourth Stage:
1. For each bounding box in third stage,
   . aspect_ratio=box_width/box_height
   . area=box_width×box_height.
2. Among all bounding boxes,
   . noisy boxes are removed by allowing boxes having specific range of aspect ratio and area

The average value of aspect ratio and the area are calculated from the generated dataset for single-line and multiline character set license plates separately.
Figure 3. 1st column: Input original image, 2nd column: Localized LP, 3rd column: Cut out of LP
7. Binarization

Documents are normally stored in gray level format having a maximum of 256 different gray values (0 to 255). To extract some information from the document image it is required to be processed number of times. Especially, if it is required to recognize the image or part of it then binary image seems to more useful than gray level image. Binarization is the method of converting any gray scale image (popularly known as multi-tone image) into black-and-white image (popularly known as two-tone image). This conversion is based on finding a threshold gray value and deciding whether a pixel having a particular gray value is to be converted to black or white. Usually within an image the pixels having gray value greater than the threshold is transformed to white and the pixels having gray value lesser than the threshold is transformed to black. The most convenient and primitive method is to find a global threshold for the whole image and binarizing the image using the single threshold. In this technique the local variations are actually suppressed or lost, though they may have important information content. On the other hand, in case of determining the threshold locally, a window is used around a pixel and threshold value is calculated for the window. Now depending on whether the threshold is to be used for the center pixel of the window or for the whole window, the binarization is done on pixel-by-pixel basis, where each pixel may have a calculated threshold value, or on region-by-region basis where all pixels in a region or window may have a threshold value.

In most of the practical cases, the binarization method fails because of the degradation of the image quality. The degradation may occur due to the poor method of acquisition of image or due to poor quality of original source. Degradation may also occur due to non-uniform illumination over the original source. The major contribution of research for binarization is to recover or extract information from a degraded image. Otsu [17] developed a method based on gray level histogram and maximizes the intra-class variance to total variance. Sauvola [18] developed an algorithm for text and picture segmentation within an image and binarized the image using local threshold. Gatos [19] used Wiener filter and Sauvola’s adaptive binarization method. In the work presented in [8] also Sauvola’s adaptive thresholding is used for binarization. Valverde [20] binarized the image using Niblack’s technique. A slight modification of Niblack’s method is done in [21] by Zhang.

In the current work Otsu’s method of binarization is applied on the selected or localized potential license plate region of the image and then binarization is done in each localized region using local threshold.

8. Character Segmentation

Segmentation is the process of isolating desired part of image from rest of the image depending on some criteria. It may be on the basis of color or grey value of the pixels within the image and even it may be depending on some other feature of the image also. In case of binarized image, segmentation indicates the isolation of foreground component from the background. In case of character segmentation, the characters constitute the foreground components. So character segmentation is basically the isolation of the characters within the image component. Connected component labeling is the process through which the characters are segmented from the background and also the individual characters are labeled distinctly to mark or identify them separately for future use. Connected Component Labeling (CCL) algorithm, or Connected Component Analysis (CCA) is an algorithmic application of graph theory, where subsets of connected components are uniquely labeled based on a given heuristic.

The outcome of CCL algorithm is a set of foreground segments which are supposed to be the characters or digits within the license plate. The set may also consist of any other foreground components within the potential license plate region. The information of each of the segments is stored in structure format consisting of label, starting row, starting column, ending row and ending column of each segment.

Figure 4(a) and Figure 4(b) show the unlabeled version of the potential license plate region before and after filtration of noise respectively. Figure 4(c) and Figure 4(d) show the potential license plate region after labeling and then after noise removal respectively. Figure 5 illustrates the retrieved character segments in the sequence of their labeling as shown in the first row and in the sequence of their starting column as shown in the second row.
Figure 4. Dat images of LP: (a) Unlabeled dat, (b) Unlabeled filtered dat, (c) Labeled dat, (d) Labeled filtered dat
9. Recognition

The segment information as provided by the CCL module is passed to the recognition module for identification of each of them in terms of a known character or digit. This module of the system is normally called Optical Character Recognition (OCR) module. There are various methods for recognition out of which the method used in this work is called “template matching” which is the most primitive and simplest method used so far. The recognition process is grouped into two sub-modules.

1. Template creation
2. Template matching

9.1 Template Creation

The characters that may be present in a license plate are from A to Z and the digits that may also be present in a license plate are from 0 to 9. So a set of 36 alphanumeric characters (26 alphabets and 10 digits) is used for template creation. The number of templates can be increased in order to increase the script-support of the OCR system. The templates are created by using the following four steps.

9.1.1 Writing the alphanumeric dataset in photo editor

Microsoft paint is used as a photo editor for this purpose. In paint each of the alphanumeric characters is written using a familiar font face and font size. Each of the images is then stored as 24-bit bmp file. Thus a set of 36 images each representing a particular template is generated.

9.1.2 Binarizing the images and cropping the character

The character images are then binarized to form black and white images. For binarizing purpose Otsu’s algorithm is used over each template image. A cropping algorithm is then applied on the binarized template images to get the images having dimension exactly equal to the character or the digit. The cropped images are then stored as 24-bit bmp file. Thus a set of 36 cropped images is generated.

9.1.3 Creating the dat image for each bmp image

Each of the binarized cropped images consists of a white background having grey value 255 and a black foreground having grey value 0. Each cropped image is then read and a corresponding dat files is generated. The dat file consists of the replica of the original image but consists of only 0’s and 1’s. Any pixel having grey value equal to 255 in the original image is mapped to ‘0’ in the dat file. Similarly, any pixel having grey value equal to 0 in the original image is mapped to ‘1’ in the dat file. The dimension of the dat file is exactly the same as its corresponding bmp file. In this way a set of 36 dat files are generated. Figure 6 shows the binarized template images and the corresponding dat images.
9.1.4 Scaling of dimension of the dat image into a standard size

For the purpose of matching of any two templates, one basic requirement is the exact matching of the dimensions of the two templates. But the original cut-outs of the template images and their corresponding dat version have the dimensions according to their own shape and size. The real time character segment also contains the dimension of arbitrary nature. Thus there is a requirement for converting the dimension of the original dat image into some standard size, known as normalization. The method used to do this is called scaling. In this work, a scaling algorithm, as described below in section 9.1.5, is written that converts any sized dat image into a standard M×N (32 × 32) dat image.

9.1.5 Algorithm for scaling the dat image

i. Set column factor $cf = N/(bc2-bc1+1)$, where $N$ is the column width of the target array and $bc2$ and $bc1$ are the ending and starting column respectively.

ii. For each row $i$, value of each column $j$, termed as $\text{picture}[i][j]$ is mapped $cf$ number of times to the target array $\text{temp}[x][y]$, where $x=i$. Any residual non-zero fractional value of $cf$ is buffered for compensation.

iii. For each row count the number of columns operated so far in the target array $\text{temp}[x][y]$.

iv. If $count! = N$ then map the last column of the row in $\text{temp}[x][y]$ with the last but one column of that particular row in $\text{temp}[x][y]$.

v. Set row factor $rf = M/(br2-br1+1)$, where $M$ is the row height of the target array and $br2$ and $br1$ are the ending and starting row respectively.

vi. For each column $j$, value of each row $i$, termed as $\text{temp}[i][j]$ is mapped $rf$ number of times to the target array $\text{picture}[x][y]$, where $y=j$. Any residual non-zero fractional value of $rf$ is buffered for compensation.

vii. For each column count the number of rows operated so far in the target array $\text{picture}[x][y]$.

viii. If $count! = M$ then map the last row of the column in $\text{picture} [x][y]$ with the last but one row of that particular column in $\text{picture}[x][y]$.

ix. A dat file is created for storing the array $\text{picture}[i][j]$.

The alphanumeric characters shown in Figure 5 in the form of dat images are normalized using the above scaling algorithm and the normalized or scaled dat images are illustrated in Figure 7. In the figure, the odd columns indicate the dat images in their original size and the even columns indicate the dat images in normalized form of size 32 × 32.
Figure 7. Segments with original size and normalized size (index 1 indicates original size and index 2 indicates normalized size)

9.2 Template matching

Matching is the last step towards recognition of characters as generated by the segmentation process. The outcome of the CCL module in terms of foreground segments is to be recognized using template matching. It means that the segments are to be converted into dat form as explained in section 9.1.3. The dat forms are to be scaled into a standard 32 × 32 size using scaling algorithm as explained in section 9.1.4 and using the algorithm 9.1.5. Now each dat segment corresponding to each character is matched with all the 36 dat templates in the library. Matching is done using two nested iterative loops, one for row scanning and another for column scanning, and comparing the position by position values of a dat segment and the templates in sequence.

A term called matching factor is computed for each pair of dat segment and template. Matching factor may be defined as a ratio of the total number of positions where the segment value matches with the template value to the total number of values in the template (i.e., 32 × 32 = 1024). Now all the templates will generate a matching factor for each segment. The matching factor may represent one of the three output status: a. exact matching, b. complete mismatching and c. confused matching. To decide whether the matching factor dictates exact matching or complete mismatching or confused matching, a confusion matrix is generated to have an idea about the matching and mismatching nature of the templates themselves. The confusion matrix has 36 rows representing each alphanumeric character and 36 columns representing also the same 36 alphanumeric characters. The value at any location (i, j) represents the matching factor of the ith template with jth template. Obviously for all i equal to j, i.e. for all the diagonal positions the matching factor become 1 as they represent the matching of the template with itself. The highest matching in any row excluding the diagonal element represents the highest ever matching of that template character with another template character within the template set. This value is denoted as mf_max.

Similarly, the smallest matching factor in any row represents the lowest ever matching of that template character with another template character within the template set. This value is denoted as mf_min.

9.2.1 Algorithm for computing elements of confusion matrix

i. Let i denotes one template and j denotes another template. N denotes the size of a template.

ii. For all i from 0 to 35 and for all j from 0 to 35 compute position by position matching of the two templates for all rows and for all columns.

iii. If n be the number of position where the two templates match each other then matching factor \( mf_{ij} = \frac{n}{N} \).

iv. For a particular i and for all j get the maximum matching factor \( mf_{\text{max}} \) and minimum matching factor \( mf_{\text{min}} \).

Now for the purpose of taking decision whether a segment generated at run time has matched to any template or not the values of \( mf_{\text{max}} \) and \( mf_{\text{min}} \) are used. A rule base is framed as described below in the form of algorithm.

9.2.2 Algorithm for matching decision

i. For a particular segment, say \( f^b \) one get the matching factors \( mf_j \) for all the templates (for all i’s).
ii. Calculate the highest matching factor \( mf_{max} \) for a particular template, say \( i^{th} \) with the \( j^{th} \) segment.

iii. If \( mf_{max} \) is greater than \( mf_{min} \), then the \( j^{th} \) segment is considered as matched with \( i^{th} \) template.

iv. If \( mf_{max} \) is lesser than \( mf_{min} \) then the \( j^{th} \) segment is considered as completely mismatched with the known character set.

v. If \( mf_{max} \) is greater than \( mf_{min} \) but lesser than \( mf_{max} \) then the \( j^{th} \) segment is considered as matched with \( i^{th} \) template with a certain level of confusion.

A sample confusion matrix is shown in Figure 8 where each row indicates a template and each column also indicates a template. The value at any location indicates the matching factor of the two templates corresponding row and column.

![Figure 8](image)

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 1 | 0.88 | 0.11 | **0.58** | 0.59 | 0.61 | 0.58 | 0.54 | 0.59 | 0.58 | 0.60 | 0.52 | 0.49 | 0.48 | 0.46 | 0.45 | 0.44 | 0.43 | 0.42 | 0.41 | 0.40 | 0.39 | 0.38 | 0.37 | 0.36 | 0.35 |
| 2 | 0.71 | 0.59 | **0.88** | 0.61 | 0.59 | 0.61 | 0.57 | 0.61 | 0.58 | 0.60 | 0.52 | 0.49 | 0.48 | 0.46 | 0.45 | 0.44 | 0.43 | 0.42 | 0.41 | 0.40 | 0.39 | 0.38 | 0.37 | 0.36 | 0.35 |
| 3 | 0.58 | 0.59 | **0.88** | 0.61 | 0.59 | 0.61 | 0.57 | 0.61 | 0.58 | 0.60 | 0.52 | 0.49 | 0.48 | 0.46 | 0.45 | 0.44 | 0.43 | 0.42 | 0.41 | 0.40 | 0.39 | 0.38 | 0.37 | 0.36 | 0.35 |
| 4 | 0.58 | 0.59 | **0.88** | 0.61 | 0.59 | 0.61 | 0.57 | 0.61 | 0.58 | 0.60 | 0.52 | 0.49 | 0.48 | 0.46 | 0.45 | 0.44 | 0.43 | 0.42 | 0.41 | 0.40 | 0.39 | 0.38 | 0.37 | 0.36 | 0.35 |

**Figure 8:** Confusion matrix for alphabets: Two boxes in a row indicate maximum (below 1) and minimum values in the row.

9.2.3 A Feedback to Localization Result

For any image if any potential license plate region is found as described in section 6 then all the succeeding procedures described in section 7 to 9.2.2 are applied on the region and after the final recognition process a recognition confidence value is measured. The recognition confidence value for the localized area is calculated just by adding all the matching factors for all the segments in the area according to the rule base described above and then dividing the value by the number of segments in the area. If a potential region has very small recognition confidence value below certain recognition threshold \((R_{thr})\) then the region is discarded to be a license plate. This recognition threshold \((R_{thr})\) is determined by adding the \( mf_{max} \) and \( mf_{min} \) for all the templates \((i=10)\) to 35) and dividing the result by 72 (36 templates and each having two extreme matching factors).

10. Experimental Results and Discussion

As the whole work is mainly divided into three stages, viz. license plate localization, character segmentation and character recognition, so the result of the current work is also observed in terms of the output of the three stages. And then the overall performance of the whole system is discussed.

Figure 3 shows the result of license plate localization. In each row the first column shows the original image, the second column shows the image with localized license plate indicated by bounding box and the third column shows the stored image containing the cut out of the localized license plate region. Because of the limitation of documentation, localization for only ten images (Figure 3-(a)-(j)) has been shown in the figure. In most of the cases the license plate is properly localized with some added region around it. This is because of the prominent edges as found in the periphery of the license plate. A careful observation shows that though Figure 3(a) contains other vehicle and also a man moving in front of the vehicle, still the method can find the license plate properly. Figure 3(c) and Figure 3(f) show the snaps taken at night. The system performs well for this case also as seen by the localization result shown in the second and third columns of the respective rows. Moreover, the images shown in Figure 3(h) and Figure 3(i) are captured during full sunlight. But the system performs well in this case also providing a good localization result. So it can well be concluded that the current work performs well throughout day and night and for variant atmospheric conditions.

Figure 4 and Figure 5 show the result of character segmentation in form of dat images. Figure 4(a) shows the unlabeled license plate and Figure 4(b) shows the unlabeled license plate after filtration. Filtration is done on the basis of some undesired pattern appearing in the unlabeled license plate, e.g. very lengthy or very thick stroke etc. Figure 4(c) shows only the labeled foreground components and Figure 4(d) shows the same after noise removal. Labeling is done while scanning the image row wise from top to bottom. So the character top appearing first is labeled first. As shown in Figure 4(d), the last ‘1’ has the highest top and so it is labeled first with ‘A’. After labeling of all the foreground components, all the characters are segmented by identifying their individual labels as shown in Figure 5.
Figure 5(a)-(i) show the segmented characters in the order of their ascending labels and Figure 5(j)-(r) show the same after rearranging them in terms of their starting columns. All these segments are normalized into a standard size of 32 × 32 and are shown in Figure 7.

Result of template matching is based on the values of confusion matrix as shown in Figure 8. Each row indicates the matching factor for the character corresponding to the row with all the characters. The two boxes in each row indicate the highest and the lowest matching factors for the character as described by $mf_{\text{max}}$ and $mf_{\text{min}}$ in section 9.2. In all the cases the value ‘1’ is not considered as it indicates the matching factor of any character with itself. For example, in first row (corresponding to the character ‘A’) the highest value is 0.71 ($mf_{\text{max}}$) in the column for ‘B’ and the lowest value is 0.50 ($mf_{\text{min}}$) in the column for ‘C’. This means that the pattern for ‘A’ is maximum matched with the pattern for ‘B’ out of all the patterns from ‘B’ to ‘Z’ and the matching ratio is 0.71. On the other hand, the pattern for ‘A’ is least matched with the pattern for ‘C’ out of all the patterns from ‘B’ to ‘Z’ and the matching ratio is 0.50. The decision rule described in section 9.2 is used for matching purpose. The template matching technique works well for most of the cases. But there are some specific characters for which most of the decisions were in confusing state and in some of the cases the result gives poor confidence value. These characters are ‘B’, ‘D’, ‘G’, ‘O’, ‘Q’, ‘U’ and the digits are ‘0’, ‘3’, ‘5’, ‘6’, ‘8’. Some forceful tuning is done on these characters depending on the positions of it in the sequence, often known as “syntax checking” of the license plate.

A quantitative measure of the performance of the proposed method can be described by considering the performance of each of the main modules and then considering the performance of the overall system. The outcome of each of the modules can be classified into three distinct categories as described below.

**Localization module**
Category A: **False negative** case if true license plate is not found and/or false locations are detected as license plate.
Category B: **False positive** case if the true license plate is found but along with that other false locations are also detected as license plate.
Category C: **True positive** case if only true license plate is detected as a license plate.

**Segmentation module**
Category A: **False negative** case if true segment is not found and/or false segments are detected as segment.
Category B: **False positive** case if the true segment is found but along with that other false segments are also detected as segment.
Category C: **True positive** case if only true segment is detected as a segment.

**Recognition module**
Category A: **False negative** case if a true character is rejected to be recognized. Category B: **False positive** case if wrong recognition is done. Category C: **True positive** case if correct recognition is done.

The result can be summarized in tabular form and is given in Table 1 below.

<table>
<thead>
<tr>
<th>Module</th>
<th>Category A</th>
<th>Category B</th>
<th>Category C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localization</td>
<td>7.50%</td>
<td>2.50%</td>
<td>90.00%</td>
</tr>
<tr>
<td>Segmentation</td>
<td>1.62%</td>
<td>1.51%</td>
<td>96.87%</td>
</tr>
<tr>
<td>Recognition</td>
<td>9.60%</td>
<td>10.20%</td>
<td>80.20%</td>
</tr>
</tbody>
</table>

The objective of the proposed work was to develop a complete ALPR system. The total system is composed of three important modules as described above. Much of the technical innovative stress is given on the localization module and to some extent in the recognition module. The CCL algorithm used for segmentation is an already known traditional method for the purpose of segmentation. Though the vertical edge detection method for localization of license plate is already reported in different literatures, the statistical distribution of the position of them in a band of rows demands some technical novelty in this context. On the other hand, in case of character recognition, though template matching is an already established primitive work, the generation of confusion matrix and thereafter the decision rule created on the basis of the matrix claims the innovation regarding the idea. The complete system is not confined in experimentation in the laboratory. Rather it is implemented as a system in a real life outdoor scenario and is run throughout the day and night. Keeping the domain of operation and the outdoor run time environment in mind and considering the simplicity of the system, the overall performance of the system is quite satisfactory.

**References**


Author Biographies

Satadal Saha has done B. Sc. (Physics Hon’s), completed B. Tech. in Applied Physics and M. Tech. in Optics and Optoelectronics from University of Calcutta in 1995, 1998 and 2000 respectively. In 2000, he joined as a Project Fellow in the Department of CST in BESU (formerly B. E. College), Howrah. In 2001, he joined as a Lecturer in the Department of Information Technology, Govt. College of Engg. and Textile Technology (formerly known as College of Textile Technology), Serampore. In 2004, he joined as a Lecturer in the Department of Computer Science and Engg., MCKV Institute of Engg, Howrah and he is continuing his service there as an Assistant Professor in the Department of Electronics and Communication. He has authored a book titled Computer Network published by Dhanpat Rai and Co. Ltd., New Delhi in 2008. His research areas of interest are image processing and pattern recognition. Mr. Saha is a member of IEEE, IETE and CSI. He was also a member of the executive committee of IEITE, Kolkata for the session 2006-08.

Subhadip Basu received his B.E. degree in Computer Science and Engineering from Kuvempu University, Karnataka, India, in 1999. He received his Ph.D. (Engg.) degree thereafter from Jadavpur University (J.U.) in 2006. He joined J.U. as a senior lecturer in 2006 and he is continuing his service there as an Assistant Professor. Currently, he is the visiting Scientist (Under BOYSCAST Fellowship from Govt. of India) at Department of ECE, The University of Iowa, USA. His areas of current research interest are OCR of handwritten text, gesture recognition, real-time image processing.

Mita Nasipuri received her B.E.Tel.E., M.E.Tel.E., and Ph.D. (Engg.) degrees from Jadavpur University, in 1979, 1981 and 1990, respectively. Prof. Nasipuri has been a faculty member of J.U since 1987. Her current research interest includes image processing, pattern recognition, and multimedia systems. She is a senior member of the IEEE, U.S.A., Fellow of LE (India) and W.B.A.S.T, Kolkata, India.