Person Identification Using OCR of Vehicle Plate

K. Sowmya Devi1, Gottipati Srinivas Babu2
PG Scholar, Dept Of Ece, Nri College Of Engineering, Agiripalli, Nunna Mandal, Krishna Dist, Andhra Pradesh, INDIA, sowmyasss68@gmail.com
Associate Professor, Dept Of Ece, Nri College Of Engineering, Agiripalli, Nunna Mandal, Krishna Dist, Andhra Pradesh, INDIA, gsrinivasbabu@yahoo.co.in

ABSTRACT
Automatic Text Recognition (ATR) is a real time embedded system which identifies the characters directly from the image of the license text area. It is an active area of research. ATR frameworks are exceptionally helpful to the law authorization organizations as the requirement for Radio Frequency Identification labels and comparative types of gear are limited. Since Text rules are not entirely drilled all over the place, it frequently ends up noticeably hard to effectively recognize the non-standard Text characters. In this paper we endeavor to address this issue of ATR by utilizing a pixel based division calculation of the alphanumeric characters in the permit content range. The non-adherence of the framework to a specific nation particular standard and textual styles adequately implies that this framework can be utilized as a part of various nations – a component which can be particularly valuable for trans-fringe activity e.g. use in nation outskirts and so on. Furthermore, there is a choice accessible to the end-client for retraining the classifier by building another example textual style database. This can enhance the framework execution and make the framework more proficient by taking pertinent examples. For greater change we creating SMS to enable activity to police in producing fines for wrong determined people in light of their portable number. This helps more in controlling the principles and directions of movement and distinguishing the mistake done individual effortlessly.

KEYWORDS: Automatic Text Recognition, License text area, radio frequency identification, characters, traffic, SMS.

INTRODUCTION
The programmed Text acknowledgment frameworks (ATR) exist for quite a while, yet just in the late 90s it turned into an imperative application in view of the extensive increment in the quantity of vehicles. The data extricated from the permit content territories is mostly utilized for activity checking, get to control, stopping, motorway street tolling, and fringe control, making auto logs for stopping frameworks, travel time estimation and so forth by the law authorization organizations. The acknowledgment issue is by and large sub-separated into 5 sections:

1. Image securing i.e. catching the picture of the permit content region
2. Pre-handling the picture i.e. standardization, changing the brilliance, skewness and differentiation of the picture
3. Localizing the permit content region
4. Character division i.e. finding and recognizing the individual image pictures on the content range
5. Optical character acknowledgment. There might be further refinements over these (like coordinating the vehicle permit number with a specific database to track speculated vehicles and so forth.) however the fundamental structure continues as before.

A controlling parameter in such manner is nation particular movement standards and models. This fines tune the framework i.e. number of characters in the permit content region, content luminance level (relative record i.e. dull content on light foundation or light content on dim foundation) and so on. So the issue would then be able to be limited for application in a specific nation. For instance, in India the standard is printing the permit content territory numbers in dark shading on a white foundation for private vehicles and on a yellow foundation for business vehicles. The general arrangement for the permit content territory is two letters (for state code) trailed by area code, at that point a four digit code particular to a specific vehicle.

Deskewing mechanism
The captured rectangular text area can be rotated and skewed in many ways due to the positioning of vehicle towards the camera. Since the skew significantly degrades the recognition abilities, it is important to implement additional mechanisms, which are able to detect and correct skewed text areas.

The fundamental problem of this mechanism is to determine an angle, under which the text area is
skewed. Then, deskewing of so evaluated text area can be realized by a trivial affine transformation. It is important to understand the difference between the “sheared” and “rotated” rectangular text area. The text is an object in three-dimensional space, which is projected into the two dimensional snapshot during the capture. The positioning of the object can sometimes cause the skew of angles and proportions. If the vertical line of text area pv is not identical to the vertical line of camera objective cv, the text area may be sheared. If the vertical lines pv and cv are identical, but the axis p a of text area is not parallel to the axis of camera c a, the text area may be rotated. (see figure 1)

Figure 1: (a) Text captured under the right angle (b) rotated text area (c) Sheared text area

2.1 Detection of skew

Hough transform is a special operation, which is used to extract features of a specific shape within a picture. The classical Hough transform is used for the detection of lines. The Hough transform is widely used for miscellaneous purposes in the problematic of machine vision, but I have used it to detect the skew of captured text area, and also to compute an angle of skew. It is important to know, that Hough transform does not distinguish between the concepts such as “rotation” and “shear”. The Hough transform can be used only to compute an approximate angle of image in a two-dimensional domain.

The mathematical representation of line in the orthogonal coordinate system is an equation \( y = a \cdot x + b \), where \( a \) is a slope and \( b \) is a y-axis section of so defined line. Then, the line is a set of all points \([x, y]\), for which this equation is valid. We know that the line contains an infinite number of points as well as there are an infinite number of different lines, which can cross a certain point. The relation between these two assertions is a basic idea of the Hough transform. The equation \( y = a \cdot x + b \) can be also written as \( a \cdot x = y - b \), where \( x \) and \( y \) are parameters. Then, the equation defines a set of all lines \((a, b)\), which can cross the point \([x, y]\). For each point in the “XY” coordinate system, there is a line in an “AB” coordinate system (so called “Hough space”)

![Figure 2: The “XY” and “AB” (“Hough space”) coordinate systems. Each point \([x_0, y_0]\) in the “XY” coordinate system corresponds to one line in the Hough space (red color). The are several points (marked as \(k, l, m\)) in the Hough space, that correspond to the lines in the “XY” coordinate system, which can cross the point \([x_0, y_0]\) .](image)

Let \( f(x, y) \) be a continuous function. For each point \((a, b)\) in Hough space, there is a line in the “XY” coordinate system. We compute a magnitude of point \((a, b)\) as a summary of all points in the “XY” space, which lie on the line \( a \cdot x < b \). Assume that \( is a discrete function, which represents the snapshot with definite dimensions \((w<h)\). To compute the Hough transform of the function like this, it is necessary to normalize it into a unified coordinate system in the following way:

\[
x' = \frac{2 \cdot x}{w} - 1; \quad y' = \frac{2 \cdot y}{h} - 1
\]

Although the space defined by a unified coordinate system is always discrete (floating point) on digital computers, we will assume that it is continuous. Generally, we can define the Hough transform \(h(a, b)\) of a continuous function \(f(x, y)\) in the unified coordinate system as:

\[
h'(a', b') = \int_{-1}^{1} f'(x', a' \cdot x + b') dx'
\]

![Figure 3: (a) Text in the unified “XY” coordinate system after application of the horizontal edge detection filter (b) Hough transform of the Text in the “B” coordinate system (c) Colored Hough transform in the “AB” coordinate system.](image)

We use the Hough transform of certain image to evaluate its skew angle. You can see the colored Hough transform on the figure 2.3.c. The pixels with a relatively high value are marked by a red color. Each such pixel corresponds to a long white line in the figure 13.a. If we assume that the angle of such lines determines the overall angle, we can find the longest line as:

\[
(a_m', b_m') = \arg \max_{0 \leq a < 1 \atop 0 \leq b < 1} \{h'(a', b')\}
\]

To compute the angle of such a line, there is a need to transform it back to the original coordinate system:

\[
[a_m, b_m] = \begin{bmatrix} w \cdot \frac{a_m' - 1}{2} \cdot h, \frac{b_m' - 1}{2} \end{bmatrix}
\]
where \( w \) and \( h \) are dimensions of the evaluated image. Then, the overall angle of image can be computed as:

\[
\theta = \arctan \left( \frac{a_m}{w} \right)
\]

The more sophisticated solution is to determine the angle from a horizontal projection of the Hough transform \( h' \). This approach is much better because it covers all parallel lines together, not only the longest one:

\[
\hat{\theta} = \arctan \left( \frac{\hat{a}'_m - 1}{2} \right) ; \quad \hat{a}'_m = \arg \max_{-\pi \leq \alpha \leq \pi}
\]

where \( p_{a a} \) is a horizontal projection of the Hough space, such as:

\[
p_{a'}(a') = \int_{-1}^{1} f'(a', b') db'
\]

### 2.4 Correction of skew

The second step of a deskewing mechanism is a geometric operation over an image \( f(x, y) \). As the skew detection based on Hough transform does not distinguish between the shear and rotation, it is important to choose the proper deskewing operation. In praxis, text areas are sheared in more cases than rotated. To correct the text area sheared by the angle \( \theta \), we use the affine transformation to shear it by the negative angle. For this transformation, we define a transformation matrix \( A \):

\[
A = \begin{bmatrix}
1 & S_y & 0 \\
S_x & 1 & 0 \\
0 & 0 & 1
\end{bmatrix} = \begin{bmatrix}
1 & -\tan(\theta) & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

where \( S_x \) and \( S_y \) are shear factors. The \( x \) \( S \) is always zero, because we shear the text area only in a direction of the \( Y \)-axis. Let \( P \) be a vector representing the certain point, such as \( P(x, y, 1) \) where \( x \) and \( y \) are coordinates of that point. The new coordinates

\[
P_z = [x_z, y_z, 1]
\]

of that point after the shearing can be computed.

where \( A \) is a corresponding transformation matrix.

Let the deskewed Text be defined by a function \( f_s \). The function \( f_s \) can be computed in the following way:

\[
f_s(z) = f \left( \frac{z}{[x, y, 1] \cdot A \cdot [1, 0, 0]^T} \right)
\]

After the substitution of the transformation matrix \( A \):

\[
f_s(x, y) = f \left( \frac{1 - \tan(\theta) x}{[x, y, 1] \cdot [0, 0, 1]^T} \right)
\]

### METHODOLOGY

The segmentation algorithm described in chapter three can sometimes detect redundant elements, which do not correspond to proper characters. The shape of these elements after normalization is often similar to the shape of characters. Because of this, these elements are not reliably separable by traditional OCR methods, although they vary in size as well as in contrast, brightness or hue. Since the feature extraction methods described in chapter four do not consider these properties, there is a need to use additional heuristic analyses to filter non-character elements. The analysis expects all elements to have similar properties. Elements with considerably different properties are treated as invalid and excluded from the recognition process.

The analysis consists of two phases. The first phase deals with statistics of brightness and contrast of segmented characters. Characters are then normalized and processed by the piece extraction algorithm.

Since the piece extraction and normalization of brightness disturbs statistical properties of segmented characters, it is necessary to proceed the first phase of analysis before the application of the piece extraction algorithm. In addition, the heights of detected segments are same for all characters. Because of this, there is a need to proceed the analysis of dimensions after application of the piece extraction algorithm.

The piece extraction algorithm strips off white padding, which surrounds the character. Respecting the constraints above, the sequence of steps can be assembled as follows:

1. Segment the text area (result is in figure 5.a).
2. Analyse the brightness and contrast of segments and exclude faulty ones.
3. Apply the piece extraction algorithm on segments (result is in figure 5.b).
4. Analyse the dimensions of segments and exclude faulty ones.

Figure 5: Character segments before (a) and after (b) application of the piece extraction algorithm. This algorithm disturbs statistical properties of brightness and contrast.
If we assume that there are not big differences in brightness and contrast of segments, we can exclude the segments, which considerably differs from the mean. Let ith segment of text area be defined by a discrete function \( f_i(x, y) \), where \( w_i \) and \( h_i \) are dimensions of the element. We define the following statistical properties of an element:

The global brightness of such segment is defined as a mean of brightnesses of individual pixels:

\[
P_{b0}^{(i)} = \frac{1}{\sum_{x=0}^{w_i} \sum_{y=0}^{h_i} f(x, y)}
\]

The global contrast of the ith segment is defined as a standard deviation of brightnesses of individual pixels:

\[
P_{b1}^{(i)} = \sqrt{\frac{1}{\sum_{x=0}^{w_i} \sum_{y=0}^{h_i} [f(x, y) - \overline{f(x, y)}]^2}}
\]

where \( \overline{f(x, y)} \) is the average brightness of the segment.

The function \( f(x, y) \) represents only an intensity of grayscale images, but the additional heuristic analysis of colors can be involved to improve the recognition process. This analysis separates character and non-character elements on color basis. If the captured snapshot is represented by a HSV color model, we can directly compute the global hue and saturation of the segments as a mean of hue and saturation of individual pixels:

\[
\overline{h(x, y)} = \frac{1}{\sum_{x=0}^{w_i} \sum_{y=0}^{h_i} h(x, y)}
\]

\[
\overline{s(x, y)} = \frac{1}{\sum_{x=0}^{w_i} \sum_{y=0}^{h_i} s(x, y)}
\]

where \( h(x, y) \) and \( s(x, y) \) is a hue and saturation of the certain pixel in the HSV color model. If the captured snapshot is represented by a RGB color model, there is need to transform it to the HSV model first. To determine the validity of the element, we compute an average value of a chosen property over all elements. For example, the average brightness is computed as

\[
\overline{p_b} = \frac{1}{n} \sum_{i=0}^{n-1} P_{b0}^{(i)}
\]

brightness is computed as \( \overline{p_b} = \frac{1}{n} \sum_{i=0}^{n-1} P_{b0}^{(i)} \), where \( n \) is a number of elements. The element \( i \) is considered as valid, if its global brightness \( p_{b0}^{(i)} \) does not differ more than 16% from the average brightness \( \overline{p_b} \). The threshold values of individual properties have been calibrated as follows:

<table>
<thead>
<tr>
<th>Property</th>
<th>Threshold Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brightness (BRI)</td>
<td>( p_{b0} = 0.16 )</td>
</tr>
<tr>
<td>Contrast (CON)</td>
<td>( p_{b1} = 0.24 )</td>
</tr>
<tr>
<td>Hue (HUE)</td>
<td>( h(x, y) = 0.15 )</td>
</tr>
<tr>
<td>Saturation (SAT)</td>
<td>( s(x, y) = 0.20 )</td>
</tr>
<tr>
<td>Width/Height Ratio</td>
<td>( w_i/h_i = 0.62 )</td>
</tr>
</tbody>
</table>

If the segment violates at least one of the constraints above, it is considered as invalid and excluded from the recognition process. The table 1 contains properties of elements. According to this table, elements 0 and 10 have been refused due to an uncommon width/height ratio, and elements 1 and 4 due to a small height.

Table 1: Properties of segments in figure. The meaning of abbreviations is as follows:

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRI</td>
<td>0.16</td>
</tr>
<tr>
<td>CON</td>
<td>0.24</td>
</tr>
<tr>
<td>HUE</td>
<td>0.15</td>
</tr>
<tr>
<td>SAT</td>
<td>0.20</td>
</tr>
<tr>
<td>WHR</td>
<td>0.62</td>
</tr>
<tr>
<td>Width</td>
<td>1</td>
</tr>
<tr>
<td>Height</td>
<td>4</td>
</tr>
</tbody>
</table>

Principle and algorithms

In some situations when the recognition mechanism fails, there is a possibility to detect a failure by a syntactical analysis of the recognized text area. If we have country-specific rules for the text area, we can evaluate the validity of that text area towards these rules. Automatic syntax-based correction of text area numbers can increase recognition abilities of the whole ATR system.

For example, if the recognition software is confused between characters „8“ and „B“, the final decision can be made according to the syntactical pattern. If the pattern allows only digits for that position, the character „8“ will be used rather than the character „B“. Another good example is a decision between the digit „0“ and the character „O“. The very small difference between these characters makes their recognition extremely difficult, in many cases impossible.

Recognized character and its cost

In most cases, characters are recognized by neural networks. Each neuron in an output layer of a neural network typically represents one character. Let \( y \) be a vector of output activities. If there are 36 characters in the alphabet, the vector \( y \) will be also 36-dimensional.

Let \( y_i \) be an ith component of the vector \( y \). Then, \( y_i \) means how much does the input character corresponds to the ith character in the alphabet, which is represented by this component. The recognized character is represented by the greatest component of the vector \( y \):

\[
\chi = \text{chr} \left( \max_{0 \leq i \leq 25} \{ y_i \} \right)
\]

where \( \text{chr}(y_i) \) is the character, which is represented by the ith component of vector \( y \). Let \( y \) be a vector descendingly sorted according to the values of components. Then, the recognized character is represented by the first component of so sorted vector:

\[
\chi = \text{chr} \left( y_0^{(5)} \right)
\]
When the recognition process fails, the first component of \( y(S) \) can contain invalid character, which does not match the syntax pattern. Then, it is necessary to use the next valid character with a worse cost.

**Syntactical patterns**

In praxis, ATR systems must deal with many different types of text area numbers. Texts are not unified, so each country has own type. Because of this, Text recognition system should be able to recognize a type of the Text, and automatically assign the correct syntactical pattern to it. The assignation of the right syntactical pattern is a fundamental problem in syntactical analysis. Syntactical pattern is a set of rules defining characters, which can be used on a certain position in a text area number. If the text area number \( P \) is a sequence of \( n \) alphanumerical characters \( P = p(0)…p(n-1) \), then the syntactical pattern \( P \) is a \( n \)-tuple of sets \( P = (p(0) \ldots p(n-1)) \), and is a set of all allowed characters for the \( i \)-th position in a text area.

For example, czech Texts can contain digit on a first position followed by a character denoting the region, where the text area has been registered and five other digits for a registration number of a car. Formally, the syntactical pattern \( P \) for czech Texts can looks like this:

\[
\begin{align*}
P &= \{0,1,2,3,4,5,6,7,8,9, C, B, K, H, L, T, N, E, P, A, S, U, L, Z, \} \\
&\quad \cap \{0,1,2,3,4,5,6,7,8,9, \} \\
&\quad \cap \{0,1,2,3,4,5,6,7,8,9, \} \\
&\quad \cap \{0,1,2,3,4,5,6,7,8,9, \} \\
&\quad \cap \{0,1,2,3,4,5,6,7,8,9, \}
\end{align*}
\]

**Choosing the right pattern**

If there are \( n \) syntactical patterns \( P^{(0)}, \ldots, P^{(n-1)} \), we have to choose the most suitable one for the evaluated text area number \( P \). For this purpose, we define a metrics (or a cost) for a computation of a similarity between the evaluated text area number and the corresponding syntactical pattern:

\[
\delta(P) = \left[ \frac{1}{\max_{0 \leq j \leq 26} \{y_j^{(i)}\}} \times 10^{-2} \right]^{10^{2}} + \sum_{i=0}^{n} \left[ \frac{1}{\max_{0 \leq j \leq 26} \{y_j^{(i)}\}} \times 10^{-2} \right]
\]

where \( \left[ \{P^{(i)}|P^{(i)} \neq P^{(i)}\} \right] \) is a number of characters, which do not match to corresponding positions in the syntactical pattern \( P \). Let \( y(i) \) be an output vector for the \( i \)-th recognized character in a text area. The greatest component of that vector \( \max_{0 \leq j \leq 26} \{y_j^{(i)}\} \) then indicates how successfully the text area has been recognized. Then, the reciprocal value of \( \max_{0 \leq j \leq 26} \{y_j^{(i)}\} \) is a cost of the character. Another way of the cost evaluation is a usage of the Smith-Waterman algorithm to compute the difference between the recognized text area number and the syntactical pattern. For example, assume that text area number ‘0B01234’ has been recognized as ‘0801234’, and the recognition pattern does not allow digit at the second position of a text area. If the character ‘8’ has been recognized with similarity ratio of 0.90, and other characters with the ratio of 0.95, the metrics for this pattern is determined as follows:

\[
\delta(P)=\left(1+\frac{10^{-2}}{0.95} \cdot 10^{-2} + \frac{10^{-2}}{0.95} \cdot 10^{-2} + \frac{10^{-2}}{0.95} \cdot 10^{-2} + \frac{10^{-2}}{0.95} \cdot 10^{-2}\right)=1.07426
\]

If there is a pattern that exactly matches to the evaluated text area number, we can say that number has been correctly recognized, and no further corrections are needed. In addition, it is not possible to detect a faulty Text, if it does not break rules of a syntactical pattern. Otherwise, it is necessary to correct detected text area using the pattern with lowest cost:

\[
P^{(sel)} = \arg \min_{P^{(sel)}} \left[ \delta(P^{(sel)}) \right]
\]

The correction of a text area means the replacement of each invalid character by another one. If the character \( i \) at the \( i \)-th position of the text area \( P \) does not match the selected pattern \( P^{(sel)} \), it will be replaced by the first valid one from \( y(s) \). \( y(s) \) is a sorted vector of output activities denoting how much the recognized character is similar to an individual character from the alphabet. Heuristic analysis of a segmented text area can sometimes incorrectly evaluate non-character elements as characters. Acceptance of the non-character elements causes that the recognized text area will contain redundant characters. Redundant characters occur usually on sides of the text area, but rarely in the middle. If the recognized text area number is longer than the longest syntax pattern, we can select the nearest pattern, and drop the redundant characters according to it.

**EXPERIMENTAL RESULTS**

![Figure 6 Template for training images](image-url)

ISSN (Online) 2278- 5841
ISSN (Print)  2320- 5156

©www.ijrcct.org | * Corresponding Author

punishable offence sms alert can be sent to a particular users due to this advantage traffic controlling made easy by identifying specific person related to the number plate

REFERENCES,

CONCLUSION
In this paper novel architecture for character recognition was tested based on Heuristic algorithm
This helps in recognition character and non characters in an image through OCR it is a process of radiant verification where typically more time compared to other techniques, but in our approach due to horizontal & vertical identification of characters through edge detection allows us to extract individual character along the number plate in less time our results exposes this particular advantage it took 0.9sec to get output as an advantage for

Figure 7 a. character identifying image using difference, b. dilated image, c. erode image, d. dilated image, e. erode image, f. Median filtered image, g. gray image, h. original image, i. Extracted image