Number Plate Detection System

Kaushik Maji, Ankur Gupta, Naveen Aggarwal
University Instt. Of Engg. & Technology,
Panjab University
Chandigarh.

Abstract

This paper presents a technique to extract registration number from the image of a moving vehicle. The system will convert the video of a moving vehicle into the sequence of images. These images are taken as input to extract the number plate of the vehicle by using the Hough transformation. The image of the number plate is further segmented in to the different characters of the registration number of the vehicle. These segmented characters are then recognized as digital characters. With the digitized number we can identify a vehicle's details easily and effectively. To improve the efficiency of the system, image of the vehicle is first enhanced by using its low level features such as contrast and brightness. Edges of the different segments of images are enhanced using the gradient features. Finally a self learning neural network is used for the effective and efficient character recognition. This technique can be deployed by Law enforcement agencies, commercial complexes and for other security purposes for example in Speed traps, Tollbooths, surveys regarding vehicles etc.

1. Introduction

Traffic management of a city has always been an astronomical task for authorities. There are many difficulties like noting the registration number of a speeding vehicle, deploying police personnel at remote locations for checking traffic violations, managing large traffic related data etc.

All these and many other issues can be resolved by building a process which can automatically detect the registration number of the vehicle. Using this registration number all the detail about the vehicle and its owner can be easily retrieved from the central database of the vehicle registration office and used in apt applications.

Such detection systems require input of a moving vehicle as video. This video is captured from high frame rate and high resolution cameras mounted on toll booths, remote areas, and other application specific locations. The images from this video are then separated to get the image containing the car.

Once the vehicle's image is fetched, it is enhanced using various image enhancement techniques for better interpretation by the system. Image is enhanced by adjusting the contrast and brightness. Further as we require only the edges or the boundaries of the number plate, gray level information in the image is sufficient. Therefore, a gray level transformation is performed. Now gradient filter is used to highlight the edges in the image.

In an image, not all the pixel contains the information at equal densities relative to the underlying application. Therefore we have to choose a certain level below which the pixels are not to be considered as useful. This is accomplished by thresholding. During the

Thresholding process, individual pixels in an image are marked as “object” pixels if their value is greater than some threshold value (assuming an object to be brighter than the background) and as “background” pixels otherwise.

Determining lines in an image require statistical analysis. This is done using Hough Transformation. Hough when applied produces a graph which on plotting shows extraordinarily bright points referencing to the lines. The parametric equations of the line obtained from the Hough plot is used to find intersections among them. These points are permuted to find a number of rectangles. The rectangle with probability of having characters and which is of a certain aspect ratio is separated out. Further noise removal is applied to enhance the extracted number plate.

The number plate is segmented to different characters. These characters are individually passed to the neural network. The self learning neural network is trained for different character fonts at earlier stage. The network outputs the appropriate Unicode value of the input character.

2. Background

The paper presents many digital image processing steps involving usage of some of the popular algorithms. This part describes them and throws light onto other available algorithms for achieving similar results.

2.1 Gradient Filters

Gradient filters are first derivative filters i.e. they are derived by differentiating f(x, y) or the image intensity function. These are particularly helpful in edge detection in images. There are many Filter operators available for usage, they are:

Sobel: The Sobel operator performs a 2-D spatial gradient measurement on both axes of an image and so emphasizes regions of high spatial frequency that correspond to edges. [1]

Laplacian: The Laplacian is a 2-D isotropic measure of the 2nd spatial derivative of an image. It highlights regions of rapid intensity change. [1]

Canny: The Canny detector is a multi-steep algorithm. It produces a resulting image where only the maximal values of the edges found are shown. [1]

2.2 Hough Transformation

[2]The Hough transform is a feature extraction technique used in image analysis, computer vision, and digital image processing. The purpose of the technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough transform. The Hough technique is particularly useful for computing a global description of a feature(s) (where the number of solution classes need not be known a priori), given (possibly noisy) local measurements. There are many variants of Hough transform being used in the market for variety of products. The simplest one is the classical method.

2.3 Character Recognition

[3]Optical character recognition (OCR) has been a topic of interest since possibly the late 1940's when Jacob Rabinow started his work in the field. The earliest OCR machines were primitive mechanical devices with fairly high failure rates. As the amount of new written material

increased, so did the need to process it all in a fast and reliable manner, and these machines were clearly not up to the task. They quickly gave way to computer-based OCR devices that could outperform them both in terms of speed and reliability. There are numerous OCR algorithms today; the basic ones however are,

Feature Point extraction:
A feature point is a point of human interest in an image, a place where something happens. It could be an intersection between two lines, or it could be a corner, or it could be just a dot surrounded by space. Such points serve to help define the relationship between different strokes. Two strokes could fully cross each other, come together in a "Y" or a "T" intersection, form a corner, or avoid each other altogether. People tend to be sensitive to these relationships; the fact that the lines in a "Z" connect in a certain way is more important than the individual lengths of those lines. These relationships are what should be used for character identification, and the feature points can be exploited for the task.

Neural networks:
[4] Neural networks are very sophisticated modeling techniques capable of modeling extremely complex functions. In particular, neural networks are nonlinear. Neural networks learn by example. The neural network user gathers representative data, and then invokes training algorithms to automatically learn the structure of the data. Although the user does need to have some heuristic knowledge of how to select and prepare data, how to select an appropriate neural network, and how to interpret the results, the level of user knowledge needed to successfully apply neural networks is much lower than would be the case using (for example) some more traditional nonlinear statistical methods.

3. Work

3.1 Technology

The project is developed on Java platform. This project uses JDK 1.6.0 and JRE 1.6.0.

3.2 JAI – Java Advanced Imaging API: Java Advanced Imaging (JAI) is a Java platform extension API that provides a set of object-oriented interfaces that support a simple, high-level programming model which allows developers to create their own image manipulation routines without the additional cost or licensing restrictions, associated with commercial image processing software. This project has been developed using JAI 1.3

3.3 Algorithms

The project has been developed in stages. These stages involve various image processing steps as described below:

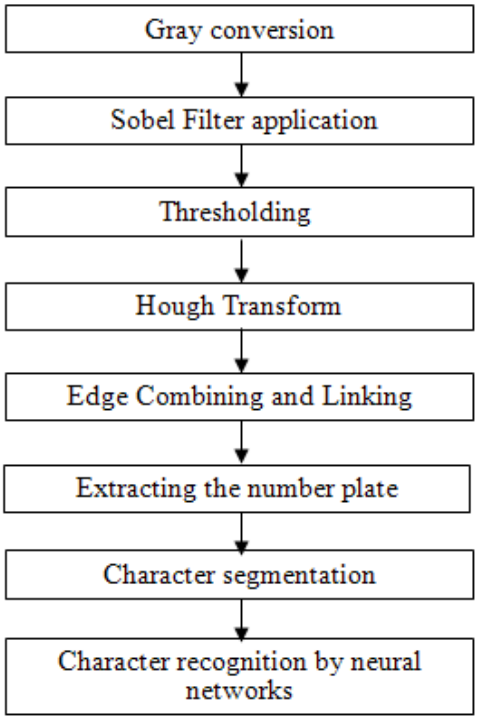


Figure 3.1

3.3.1 Gray conversion

Since the intensity values of the image are sufficient for the subsequent steps, the RGB image is converted to Gray image.

3.3.2 Sobel Filter

The Sobel operator is used in image processing, particularly within edge detection algorithms. Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Sobel operator is either the corresponding gradient vector or the norm of this vector. The Sobel operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image.

In simple terms, the operator calculates the gradient of the image intensity at each point, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The result therefore shows how "abruptly" or "smoothly" the image changes at that point and therefore how likely it is that that part of the image represents an edge, as well as how that edge is likely to be oriented. In practice, the magnitude (likelihood of an edge) calculation is more reliable and easier to interpret than the direction calculation.

Mathematically, the gradient of a two-variable function (here the image intensity function) is at each image point a 2D vector with the components given by the derivatives in the horizontal and vertical directions. At each image point, the gradient vector points in the direction of largest possible intensity increase, and the length of the gradient vector corresponds to the rate of change in that direction. This implies that the result of the Sobel operator at an image point which is in a region of constant image intensity is a zero vector and at a point on an edge is a vector which points across the edge, from darker to brighter values.

3.3.3 Thresholding

The filtered image is converted into binary image by choosing a appropriate threshold value so that redundant pixels are removed. Thresholding value depends on the lighting in the image and will vary from image to image. In other words, low magnitude edges in the image are removed.

3.3.4 Hough Transformation

The classical Hough transform was concerned with the identification of lines in the image, but later the Hough transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses.

For generating the Hough transform of the image, every pixel taken into consideration (i.e. the bright ones) after Thresholding are put in the parametric equation of line as shown below:

$$R = X \cos\emptyset + Y \sin\emptyset \qquad \text{Eq. (3.1)}$$

Here the value of Ø is varied from 0 to 360 degrees to obtain the value of R for the same pixel. In other words the point is considered to be part of all lines having value of Ø in the above integral range. This gives a sine like curve for every pixel considered. The brighter points in the resulting image are indicators of lines present in the actual image. The following is the Hough transform obtained from a car's image.

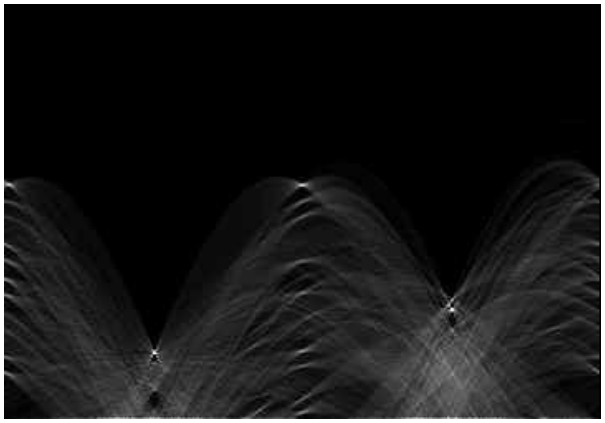


Figure 3.2

3.3.5 Edge combining and Linking

After edge detection using Sobel filter, the image is added with its gray image. The result of this combination or merging is an Image with prominent edges. This image is again used for a better Hough transformation. The combination is done using an add operator of JAI.

3.3.6 Selection of required lines

The bright points in the Hough correspond to r and theta value of the edge lines in parametric form. These values are used to map the lines onto the binary image obtained after the thresholding. There would be multiple wanted and unwanted lines. The unwanted lines are rejected or filtered out by using the value of theta which is predefined. These predefined values are solely dependent on the orientation of the camera. In this paper we would take the orientations to be 90, 180,270 or 360 as Ø values. These correspond to horizontal and vertical lines only.

3.3.7 Identification of number plate

The differences between each horizontal and vertical line with other horizontal and vertical lines respectively are evaluated and stored. Those pairs of horizontal and vertical lines are selected whose ratio of differences corresponds to the defined aspect ratio. This gives the lines corresponding to the number plate.

The intersection points of these lines are calculated using parametric equation of the line. These points can correspond to multiple rectangles depending upon other particulars of the image. The desired Rectangle is chosen by considering a single line through the middle of each one of them. A counter counts the number of intersections that the considered middle line has for each rectangle. The rectangle with highest number of intersections has the highest probability of having characters and hence is chosen as the number plate.

3.3.8 Extraction of number plate

The points obtained in the previous process are passed to the cropping function which extracts the desired rectangle. This is the required number plate.

3.3.9 Character segmentation

After the Number plate extraction, noise is removed from the plate. For this we again use Thresholding. This gives a binary image as output having only the characters in dark and background cleared.

Now vertical scanning is used to get the number of lines in which the characters are written. For this we start from the top and check for a horizontal line having a black pixel and then we move on until we find another horizontal line which has no black pixel. This gives us the width of the first line with characters. In this manner other lines are found too. Then characters are obtained in a similar manner for every line found, starting from the left and checking for the first vertical line having black pixel and stopping at the line where no black pixel is obtained. After that next character is found in the same line and so on. These coordinates are passed to the

cropping function to crop and store the characters as individual images.

3.3.10 Character recognition

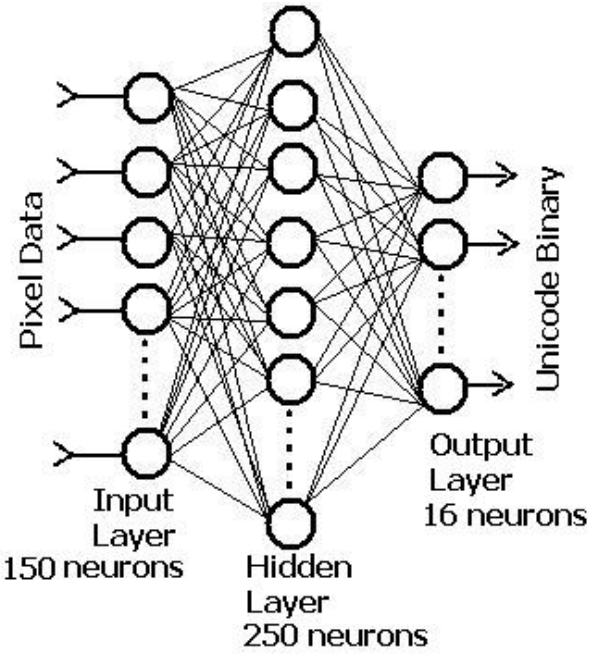
This is the final step and involves images of characters as input. The images of characters stored in the previous step are mapped into an image having dimension, 15×10. These 150 pixels acts as the input to a Multi layered perceptron (MLP) Neural network.

[4] The Multi-Layer Perceptron Neural Network is perhaps the most popular network architecture in use today. The units each perform a biased weighted sum of their inputs and pass this activation level through an activation function to produce their output, and the units are arranged in a layered feed forward topology. The network thus has a simple interpretation as a form of input-output model, with the weights and thresholds (biases) the free parameters of the model. Such networks can model functions of almost arbitrary complexity, with the number of layers, and the number of units in each layer, determining the function complexity. Important issues in Multilayer Perceptrons (MLP) design include specification of the number of hidden layers and the number of units in each layer. Most common activation functions are the logistic and hyperbolic tangent sigmoid functions.

The next part of the project discussed is still under development. The MLP Network for the purpose of this project would be composed of 3 layers, one input, one hidden and one output.

The input layer will be constituted by 150 neurons which will receive pixel binary data from a 10x15 symbol pixel matrix. The size of this matrix was decided taking into consideration the average height and width of character image that can be mapped without introducing any significant pixel noise.

The hidden layer will have 250 neurons. The output layer would be composed of 16 neurons corresponding to the 16-bits of Unicode encoding.



To initialize the weights a random function would be used to assign an initial random number which lies between two preset integers named ±weight bias.

To train the network, trainer images corresponding to various font types would be used. These would be input to the network to generate weight values for connections in the network. On obtaining a different output than desired, the error would be noted and proper corrections would be made to account for the same.

The images of characters obtained after segmenting would then be passed to the neural network and a 16 bit output will be generated. This output will be inferred as the Unicode

value of the character to be recognized. Now the Unicode would simply be converted to the corresponding character.

4. Results

At this stage we are able to extract the number plate from a given car's image and segment characters from it (considering a particular orientation of the camera). The following test case clearly depicts our work:

Figure 4.1

Step 1 - Source image

Figure 4.2

Step 2 - Taking half of the image from bottom

Figure 4.3

Step 3 - RGB to Gray conversion

Figure 4.4

Step 4 - Application of SOBEL filter

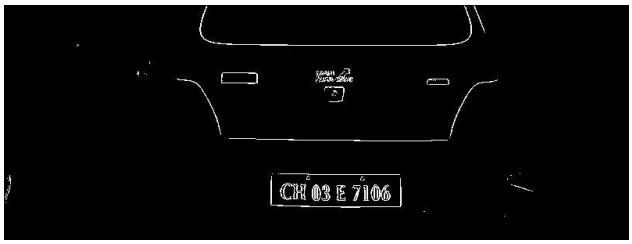


Figure 4.5

Step 5 - Application of Thresholding

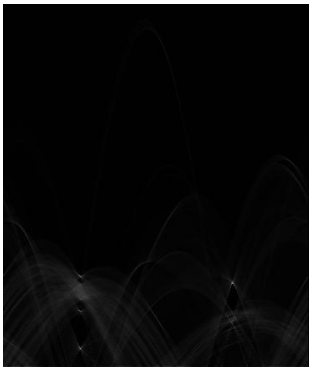


Figure 4.6

Step 6 – Hough Transform

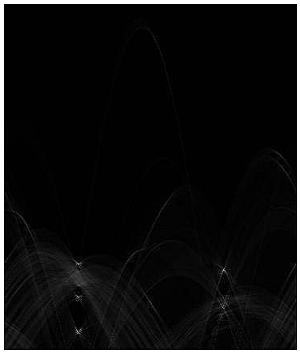


Figure 4.7

Step 7 – After convolving the Hough Transform, to get better bright spots.

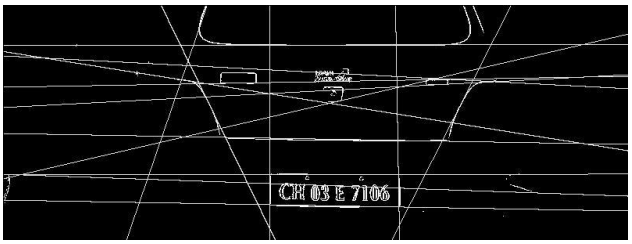


Figure 4.8

Step 8 - Mapping the lines on the image as suggested by the Hough Transform.

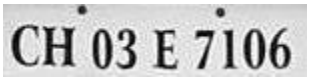


Figure 4.9

Step 9 - Extracted Number plate.

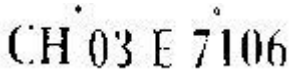


Figure 4.10

Step 10 - Background Removed.

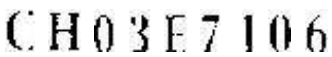


Figure 4.11

Step 11 - Segmented characters

5. Conclusion

The images we took were clear and had easily recognizable features. The application could easily extract out the exact number plate. But there are certain issues at this point of time as the project advances:

- The threshold value has to be manually chosen. Thus an automated solution based on Image statistics is required.
- Parameters selected are based on how and where the camera is being mounted.
- Environmental characteristics are also very important. Due these effects, the input images may get noisy. E.g. Fog, smog etc which lowers human visibility would also inhibit the systems functionalities.

6. References

[1] Reference Manual for the TNT products V6.30. / 3461.2

[2]D. Ballard and C. Brown Computer Vision, Prentice-Hall, 1982.

[3]Character Recognition by Feature Point Extraction, by Eric brown

[4]Unicode Optical Character Recognition
By Daniel Admassu