Development and Testing of ANPR System

K. Sivamani[1] , D. Nedumaran[2]
[1, 2] Department of Central Instrumentation & Service Laboratory
[1, 2] University of Madras, Guindy Campus, Chennai , TN, India

Abstract- *This paper deals with the development of Automatic Number Plate Recognition (ANPR) system to identify the vehicle registration number. In this work, kNN based algorithm for plate localization and recognition of various font styles in the number plates was proposed. Initially, the vehicle's number plate image was acquired through webcam and preprocessed using RGB to HSV transformation and Gaussian blur filter, which removed the blurring artifacts in the image. Then, the plate localization was done by using morphology transformation algorithm and then the Region of Interest (ROI) of the Number Plate was extracted. Extractions of number plate's characters were segmented using OTSU thresholding algorithm. Finally, an appropriate kNN based Optical Character Recognition (OCR) algorithm retrieved the characters from the number plate. Experimental results exhibited that the proposed algorithm well identified the vehicle's number irrespective of the varying illumination and changing background conditions. Finally, the resultant number plate characters and related images were stored in database with owner details for future processing and testing.*

Keywords- pre-processing; number plate localization; character segmentation; OCR; OTSU.

I. INTRODUCTION

Automatic number plate recognition (ANPR) is a technology that uses optical character recognition. ANPR is mainly used in images to read the registration plate of the vehicle. Usually, ANPR system captures and stores the images using cameras and extracts the text from the number plate for the recognition of the details of the vehicle concerned. For processing and extraction of the details of the vehicle, an application program installed in the computer performs various image processing operations, like image enhancement, segmentation, feature extraction, and character recognition. In a typical ANPR system, the image manipulation technique has been used to detect the numbers from the normalized and enhanced image of the number plate. Then, optical character recognition (OCR) has been performed to extract the alpha numeric characters in the number plate. Two basic approaches have been practiced in the ANPR systems. Firstly, the entire process will be performed at the lane location in real time. Secondly, several images from many lanes will be transmitted to a remote computer, where the OCR process will be performed at some later point of time [1].

II. LITERATURE REVIEW

In the past decades, several methods have been proposed for ANPR systems that consist of pre-processing, number plate localization (NPL), character segmentation (CS) and optical character recognition (OCR) techniques. These methods involve simple to sophisticated complex mathematical methods. Some of the basic techniques used for pre-processing are gray scale conversion [2], skew correction and normalization [3], Gaussian smoothing [4], Median filter [5], and RGB to YUV conversion [6]. Some of the distinctive techniques available for NPL detection are histogram [7], fuzzy logic [8], Haar-like classifiers [9], Wavelet based detection [10], Rectangle/symbol search [11], etc. Methods used for CS are Blob analysis [12], Connected Component Analysis (CCA) [13], Horizontal/ vertical projection [2], Watershed algorithm [14], Edge detection [15], etc., For OCR, pattern matching [2], four classifiers [3], Artificial Neural Network (ANN) [16, 17], and Support Vector Machine (SVM) classifier [5] techniques have been testified. In this work, a novel hybrid technique employing morphological transformation for localization, OTSU for segmentation and kNN for character recognition was attempted.

III. PROPOSED SYSTEM

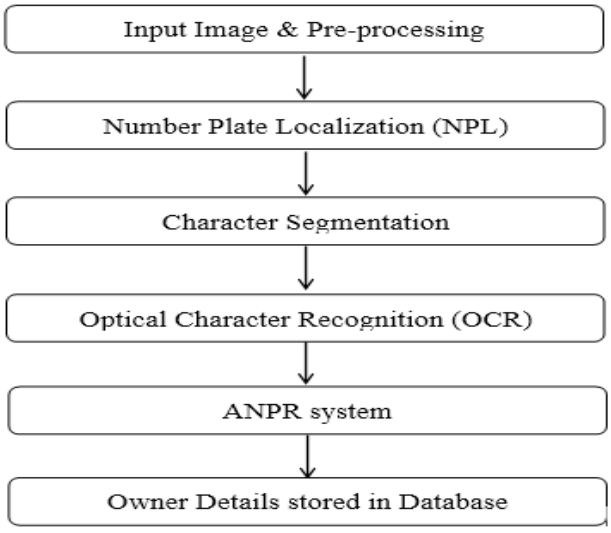

Figure 1. Flowchart of the ANPR algorithm

The flowchart of the proposed ANPR system using kNN based character recognition is shown in Figure 1.

A. Pre-processing

In pre-processing the image is improved by removing the data that suppresses unwanted distortions or enhancing some image features which are very important for further processing. The geometric transformations of images like rotation, scaling, and translation are classified among pre-processing methods[17]. Here, pre-processing consists of two steps.

1) Raw RGB image conversion using HSV transformation.
2) Gaussian blur filter for removing the noise and improving better smoothing.

The RGB color model is an additive primary color model in which red, green and blue light are added together in various ways to reproduce a broad array of colors. HSV stands for hue, saturation, and value, and is also often called HSB (B for brightness) [18]. In case of 8-bit and 16-bit images, R, G, and B are converted to the floating-point format and scaled to fit the 0 to 1 range using the following Eq.(1)

$$V \leftarrow \max(R,G,B)$$

$$S \leftarrow \begin{cases} \dfrac{V-\min(R,G,B)}{v} & if\ V \neq 0 \\ 0 & otherwise \end{cases}$$

$$H \leftarrow \begin{cases} 60(G-B)/(V-\min(R,G,B)) & \text{if V = R} \\ 120+60(B-R)/(V-\min(R,G,B)) & \text{if V = G} \\ 240+60(R-G)/(V-\min(R,G,B)) & \text{if V = B} \end{cases} \quad \ldots(1)$$

If $H < 0$, then $H \leftarrow H + 360$.

The output should be $0 \le V \le 1, 0 \le S \le 1, 0 \le H \le 360$

1) Gaussian blur filter

A Gaussian blur is the result of blurring an image by a Gaussian function. The Gaussian blur is also known as Gaussian smoothing. Applying a Gaussian blur has the effect of reducing the high-frequency component of an image. As a result a Gaussian blur is thus act like a low pass filter. The Gaussian blur is a type of image-blurring filter which uses a Gaussian function for calculating the transformation applied to each pixel in the image. The one dimensional equation of a Gaussian function is

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-x^2}{2\sigma^2}} \quad \ldots(2)$$

In two dimensions, it is the product of two such Gaussians, one in each dimension and is given by the Eq.(3).

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-x^2+y^2}{2\sigma^2}} \quad (3)$$

where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis and σ is the standard deviation of the Gaussian distribution. When the Gaussian function is applied in two dimensions, the above formula produces a surface of contours, which are concentric circles with a Gaussian distribution from the center point. The values obtained from this distribution are used to build a convolution matrix and it is applied to the original image [19].

B. Number Plate Localization (NPL)

Plate localization is responsible for finding and isolating the plate on the picture. The image acquired from the camera was converted into RGB to HSV image and then the localization of the number plate was done by advanced Morphological Transformation algorithm.

Here, OpenCV function morphologyEx was used to perform the morphological transformation operations such as:

i. Opening
ii. Closing
iii. Morphological Gradient
iv. Top Hat
v. Black Hat

(i) Opening

Opening is obtained by the erosion of an image, and then followed by dilation. It is used in the removal of small objects (it is assumed that the objects are bright on a dark foreground).
dst = open(src,element) = dilate(erode(src,element))

(ii) Closing

Closing is obtained by the dilation of an image followed by erosion and it is useful to remove small holes (dark regions).
dst = close(src,element) = erode(dilate(src,element))

(iii) Morphological Gradient

It is the difference between the dilation and the erosion of an image. It is useful for finding the outline of an object.

dst= morph_grad(src,element) = dilate(src,element) – erode(src,element)

(iv) Top Hat

Top Hat transform is the difference between an input image and its opening.

dst=tophat(src,element) = src – open(src,element)

(v) Black Hat

Black Hat transform is the difference between the closing and its input image [20].

dst= blackhat(src,element) = close(src,element) – src

C. Character segmentation

Character segmentation was done by using OTSU threshold algorithm. In image processing and computer vision applications, OTSU perform the clustering based image thresholding automatically, otherwise, the reduction of a gray level image into a binary image has to be done. In this algorithm, it assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels). It calculates the optimum threshold value for separating the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal.

a) Procedure

In Otsu's method we searched for the threshold which minimizes the intra-class variance i.e., the variance within the class, which is defined as a weighted sum of variances of the two classes and is given by the Eq.(4)

$$\sigma_{\omega}^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \quad (4)$$

Weights ω_0 and ω_1 are the probabilities of the two classes separated by a threshold t, σ_0^2 and σ_1^2 are variances of these two classes. The class probability of $\omega_{0,1}(t)$ is computed from the L bins of the histogram:

$$\omega_0(t) = \sum_{i=0}^{t-1} p_{(i)} \quad (5)$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p_{(i)} \quad (6)$$

Otsu shows that minimizing the intra-class variance is the same as maximizing inter-class variance:

$$\sigma_b^2(t) = \sigma^2 - \sigma_{\omega}^2(t) = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2$$

$$= \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2 \quad (7)$$

The Eq.(7) is expressed in terms of class probabilities ω and class means μ.

Whereas, the class mean $\mu_{0,1,T}(t)$ is given by,

$$\mu_0(t) = \sum_{i=0}^{t-1} i\frac{p(i)}{\omega_0} \quad (8)$$

$$\mu_1(t) = \sum_{i=t}^{L-1} i\frac{p(i)}{\omega_1} \quad (9)$$

$$\mu_T = \sum_{i=0}^{L-1} ip(i) \quad (10)$$

Therefore, the class mean can be related and verified using the following Equations.

$$\omega_0\mu_0 + \omega_1\mu_1 = \mu_T \quad (11)$$

$$\omega_0 + \omega_1 = 1 \quad (12)$$

The class probabilities and class means can be computed iteratively. This idea yields an effective algorithm.

A. *Algorithm*

The algorithm comprises of the following six steps [21]:

1. Compute histogram and probabilities of each intensity level
2. Set up initial $\omega_i(0)$ and $\mu_i(0)$
3. Step through all possible thresholds t=1, 2, 3. …. maximum intensity
4. Update ω_i and μ_i
5. Compute $\sigma_b^2(t)$
6. Desired threshold corresponds to the maximum $\sigma_b^2(t)$.

D. Optical Character Recognition (OCR)

After the segmentation of elements (characters and numbers), the final module in the number plate recognition process is character recognition. OCR was done by using kNN algorithm. First the system was trained by set of samples of characters array which is called training data.

a) Training data

Training data includes several components: every training sample is a vector with the values, which is sometimes referred to as feature vector in computer vision. Generally, all the vectors have the same number of components i.e., features; In this module, assumes that each of the feature can be ordered and its values are floating-point numbers, which can be compared with each other and strictly ordered. Optional set of responses corresponds to the samples. Generally, the responses are scalar values and are ordered or categorical, when we dealt with classification problem. In this case, the responses are often called "labels" [22].

b) Nearest neighbor

It is the simplest of the entire machine learning algorithm. This is mainly based on the idea, to memorize the training set, and to predict the label of new instance on the basis of labeling the closest neighbors in the training set. This is based on the assumption that the domain points are described by the features which are relevant to their labeling. The labeling is made in such a way that the closest point and likely point have the same label. For some reason, if the training set is immense, then finding the nearest neighbor will be done very soon.

c) kNN algorithm for OCR

The k-Nearest Neighbors algorithm (kNN) is a non-parametric method and used for classification and regression. The input consists of the k closest training examples in the feature space in both cases. The output depends on classification or regression used by kNN. The training examples are vectors in a multi-dimensional feature space. Each feature space is with a class label. The training phase of the algorithm consists of only storing feature vectors and the class labels of the training samples. Here, k is a user-defined constant in the classification phase, and an unlabeled vector (a query or test point) classified by assigning the label. The label is the most frequent among the k training samples nearest to that query point. Euclidean distance is the commonly used distance metric for continuous variables. For discrete variables, such as for text classification, another metric can be used, such as the overlap metric (or Hamming distance) [23].

d) k-Nearest Neighbor Estimator

The algorithm takes all training samples and predicts the response. This response is for the new sample. By analyzing a certain number (k), the nearest neighbors of the sample is estimated from the weighted sum. This method is sometimes referred as "learning by example" because for prediction it looks for the feature vector with a known response that is closest to the given vector.

Consider S = (x_1, y_1), …… , $(x_m; y_m)$ be a sequence of training samples. For each $x \in X$, let $\pi_1(x),\ldots\ldots, \pi_m(x)$ be a reordering of $\{1,\ldots\ldots,m\}$ samples according to their distance to x, $\rho(x, x_i)$. That is, for all i < m,

$$\rho(x, x_{\pi\, i(x)}) \leq (x, x_{\pi\, i+1(x)}) \tag{13}$$

For a number k, the kNN rule for binary classification is defined as follows:

- Input: a training sample S = $(x_1, y_1),\ldots\ldots, (x_m; y_m)$
- Output: for every point $x \in X$, return the majority label among $\{y_{\pi i(x)} : i \leq k\}$
- When k = 1, we have the 1-NN rule:

$$h_s(x) = y_{\pi 1}(x) \tag{14}$$

- For regression problems, namely, Y = R, one can define the prediction to be the average target of the k nearest neighbors. That is,

$$h_s(x) = \frac{1}{k}\sum_{i=1}^{k} y_{\pi_i}(x) \tag{15}$$

- More generally, for some function $\phi : (X \times Y)^k \rightarrow Y$, the k-NN rule with respect to ϕ is [24]:

$$h_s(x) = \phi((x_{\pi 1}(x), y_{\pi 1}(x)),\ldots\ldots\ldots,(x_{\pi k}(x), y_{\pi k}(x))) \tag{16}$$

- For example, if Y = R, we can take a weighted average of the targets according to the distance from x:

$$h_s(x) = \sum_{i=1}^{k} \frac{\rho(x, x_{\pi i}(x))}{\sum_{j=1}^{k} \rho(x, x_{\pi_j}(x))} y_{\pi i}(x) \tag{17}$$

IV. SIMULATION RESULT AND DISCUSSION

In this work, all the algorithms of the ANPR system were developed in Open Source Computer Vision (OpenCV) - a free open-source BSD licensed software tool, which consists of predefined machine learning and computer vision libraries. It is used for developing algorithms for ANPR system. In this work, C++ language was used for developing the programs. The ANPR algorithm development and its testing are procedures are given in the following sub-sections.

A. Generating the training data

A set of characters array was written as XML file format and called for training and testing the machine. The output of generating the training data is shown in the Fig. 2.

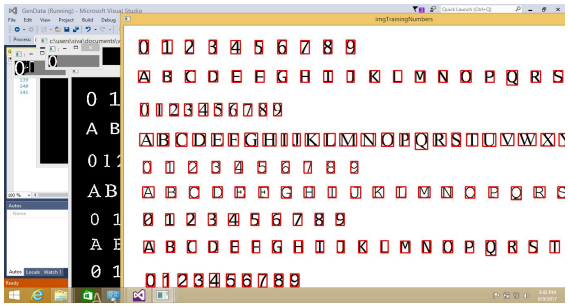

Figure 2. Output of generated Training data

B. Train & test OCR

The set of characters array was generated and trained as shown as above. Now the trained data was tested with some characters set. The resultant output is shown in Fig. 3.

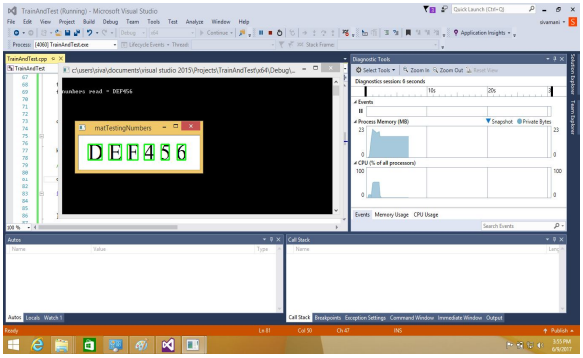

Figure 3. Output of train & test OCR

C. Pre-processing

In figure 4, the left side mage is the original raw RGB image, which was transformed to HSV image using transformation. The output of the transformation is shown in the right side of Fig. 4.

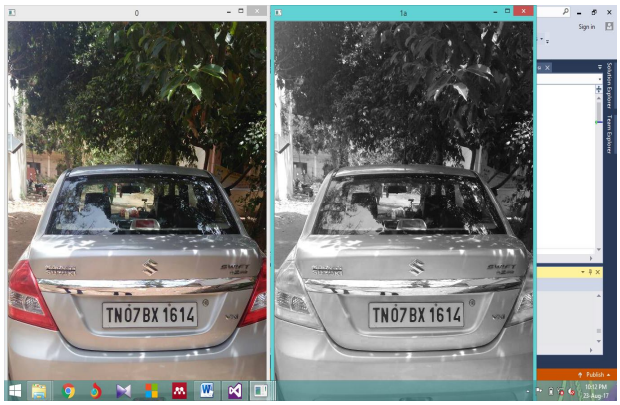

Figure 4. Raw image of RGB converted into HSV image

D. Gaussian blur algorithm

The HSV transformed image was subjected to another preprocessing step, called Gaussian blur algorithm. The fifth iteration of the Gaussian blur algorithm and its corresponding output obtained is shown in the Fig. 5.

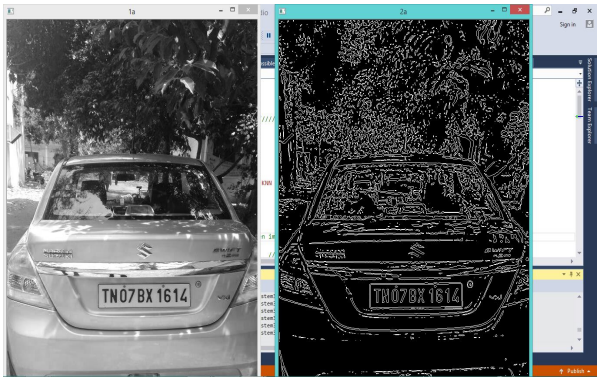

Figure 5. Gaussian blur filter after better smoothing and removal of noise

E. Plate localization

The plate localization was done by using advance morphology transformation algorithm. In this stage, the number plate was extracted for further processing. Using this algorithm, the number plate was localized as shown in the Fig.6.

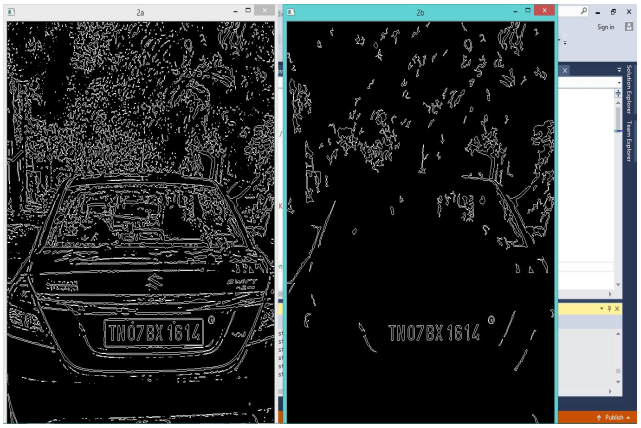

Figure 6. Plate localized image after applying advanced morphological transformation algorithm

F. Character segmentation

The segmentation of the characters in the number plate was performed using OTSU thresholding algorithm. The localized number plate was segmented and the character segmented image is shown in the Fig. 7.

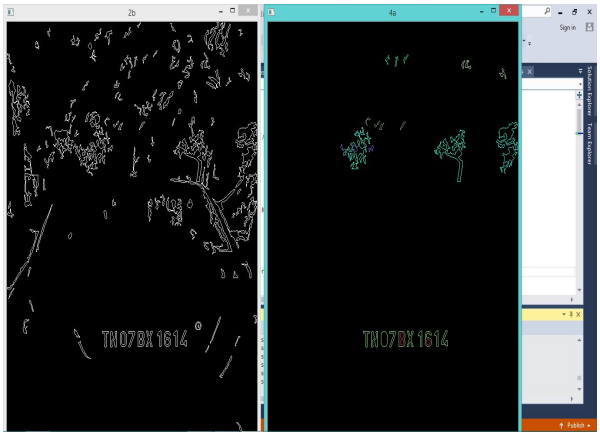

Figure 7. Character Segmented image after applying OTSU thresholding algorithm

G. OCR

The final stage of the ANPR system is the OCR. The characters are recognized using kNN algorithm. Using the kNN algorithm the characters are recognized and matching with the training data. The resultant output obtained in the real-time ANPR system using webcam is shown in the Fig. 8.

Figure 8. ANPR system

E. Parameters for ANPR system

To estimate the performance of the Gaussian blur filter with other traditional filter, a comparative study was conducted and the various parameters estimated for comparison is given in Table I.

Fig 9 shows the histogram plot of the performance metrics of the proposed Gaussian blur filter against the traditional filters. From this study, the Gaussian Blur Filter is outperformed over the other traditional filters in terms of the estimated performance metrics given in Table I.

Table 1. Performance study of Gaussian Blur Filter

PARAMETERS	GAUSSIAN BLUR	BILATERAL BLUR	HOMOGENOUS BLUR	MEDIAN BLUR
Peak Signal to Noise Ratio (PSNR)	38.7379	38.6637	38.6637	36.3512
Average Deviation (AD)	1.1597	1.4839	1.4839	2.0321
Mean Square Error (MSE)	8.6955	8.8454	8.8454	15.0647
Normalized absolute error	0.0157	0.0202	0.0202	0.0279
Normalized Cross Correlation	0.9946	0.9953	0.9953	1.0029

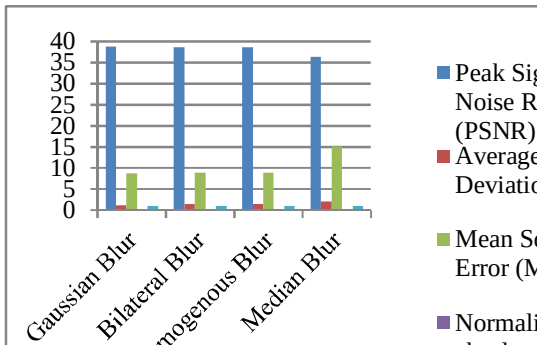

Figure 9. Histogram plot of the performance metrics of the proposed Gaussian Blur Filter

F. Database development

After the recognition of characters in ANPR system, the acquired image and related data are stored in database using C# on visual studio IDE platform. MS Access was used to tabulate the details and MySQL language for query. The database for vehicle owner details using C# is shown in Fig. 10.

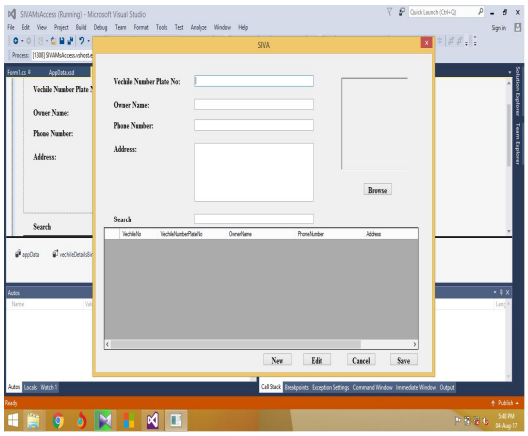

Figure 10. Database having three vehicles owner details

Initially, three vehicle owner details were generated and stored in a database for testing the algorithm and verification of results obtained. The snapshot of the vehicle owner details verification is shown in the Fig. 11.

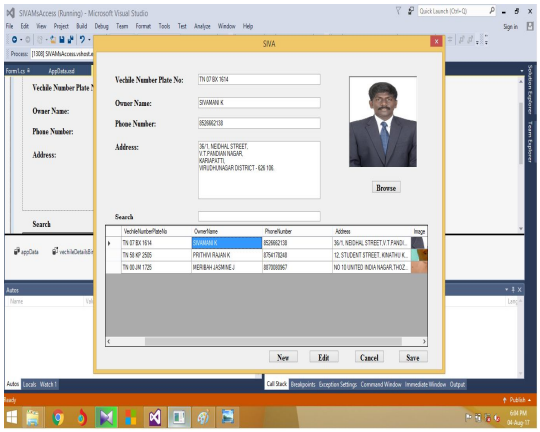

Figure 11. ANPR system with Database verification

V. CONCLUSION

In this work, the efficient real time ANPR system was developed using kNN algorithm. Before applying the kNN algorithm, the image acquired through the webcam and the databases were preprocessed using Gaussian Blur filter for removing the noise. The performance of the Gaussian Blur filter was studied by estimating the performance metrics and compared with other traditional filters. Further, the plate localization was carried out using advanced morphology transformation algorithm. Then, the character segmentation was implemented using OTSU thresholding algorithm. Finally, the OCR was experimented using kNN algorithm. Using the developed ANPR algorithm, a database was developed to store the details of the owner as well as the related images for future use. Experimental results exhibited that the proposed algorithm well identified the vehicle's number irrespective of the varying illumination and changing background conditions. Further, the results of this study revealed that the developed ANPR system very well detected the number plate without any error and detected the owner's detail in real-time. As a result, the proposed ANPR system can be deployed in real-time after undergoing field trails, standardization and expert opinion.

VI. FUTURE SCOPE

Presently, the ANPR system was implemented in the software environment, which requires a PC and related imaging equipment. Further, the execution time does not meet the real-time requirements of the system. To address these issues, the ANPR system will be implemented in the hardware environment like FPGA (Field Programmable Gate Array) in the future.

REFERENCES

[1] Automatic number plate recognition:

https://en.wikipedia.org/wiki/Automatic_number_plate_recognition

[2] Choudhury A. Rahman, Wael Badawy and Ahmad Radmanesh, "A Real Time Vehicle ' s License Plate Recognition System ", Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.

[3] Xiang Pan et.al., "A hybrid method for robust car plate character recognition", Engineering Applications of Artificial Intelligence, Elsevier Ltd., pp. 943-972, 2005.

[4] H.Erdinc Kocer and K.Kursat Cevik,"Artificial neural networks based vehicle license plate recognition", Procedia Computer Science, Published by Elsevier Ltd., pp. 1033-37, 2010.

[5] Runmin Wang et.al., "License plate detection using gradient information and Cascade detectors ", Optik, Elsevier GmbH, pp. 186-190, 2013.

[6] Muhammad Rizwan Asif et.al., "Multinational vehicle license plate detection in complex backgrounds", J. Vis. Commun. Image R., Elsevier Inc., pp. 176-186, 2017.

[7] Z. Baohua, Y. Dahua, H. Hongmei, and G. Lanying, "License Plate Location Algorithm Based on Histogram Equalization," IEEE International Conference on Computer Design and Applications (ICCDA), vol. 1, no. Iccda, pp. 7–9, 2010.

[8] C. N. Paunwala, S. Patnaik, and C. Engineering, "An adaptive integrated rule-based algorithm for license plate localization," Opto- Electronics Review, Springer, vol. 20, no. 4, pp. 323–334, 2012.

[9] Lihong Zheng et.al., "An algorithm for accuracy enhancement of license plate recognition", Journal of Computer and System Sciences, Elsevier Inc., pp. 245-255, 2012.

[10] C. N. Paunwala and S. Patnaik, "An Improved License Plate Extraction Technique Based on Gradient and Prolonged Haar Wavelet Analysis," International Conference and Workshop on Emerging Trends in Technology, no. Icwet, pp. 618–622, 2010.

[11] N. F. Gazc´on, C. I. Ches˜nevar, and S. M. Castro, "Automatic vehicle identification for Argentinean license plates using intelligent template matching," Pattern Recognition Letters,Elsevier, vol. 33, pp. 1066–1074, jul 2012.

[12] Y. Yoon, K.-d. Ban, H. Yoon, and J. Kim, "Blob Detection and Filtering for Character Segmentation of License Plates," IEEE 14th International Workshop on Multimedia Signal Processing (MMSP), pp. 349–353, 2012.

[13] Tejendra Panchala et.al., "License Plate Detection using Harris Corner and Character Segmentation by Integrated Approach from an Image", 7th International Conference on Communication, Computing and Virtualization 2016, Procedia Computer Science, Elsevier B.V, pp. 419-425, 2016.

[14] H.-m. Interface, L. Chao-yang, and L. Jun-hua, "Vehicle License Plate Character Segmentation Method Based on Watershed," IEEE International Conference on Machine Vision and Human-machine Interface Vehicle, pp. 447–452, 2010.

[15] Pratiksha Jain et.al., "Automatic License Plate Recognition using OpenCV", International Journal of Computer Applications Technology and Research, vol.3, pp. 756-761, 2014, Issue 12, 756 - 761, 2014, ISSN:-2319–8656.

[16] Jianbin Jiao et.al., "Aconfigurable method for multi-style license plate recognition", Pattern Recognition, Elsevier Ltd., pp. 358-369, 2008.

[17] M. Sonka et al., "Image Processing, Analysis and Machine Vision", 1993, pg.56, ISBN-10: 049508252X, ISBN-13: 978-0495082521.

[18] Color Models: RGB, HSV, HSL: https://en.wikibooks.org/wiki/Color_Models:_RGB,_HSV,_HSL

[19] Gaussian blur: https://en.wikipedia.org/wiki/Gaussian_blur

[20] More Morphology Transformations: http://docs.opencv.org/2.4/doc/tutorials/imgproc/opening_closing_hats/opening_closing_hats.html

[21] Otsu's method: https://en.wikipedia.org/wiki/Otsu%27s_method

[22] Training Data — OpenCV 3.0.0-dev documentation: http://docs.opencv.org/3.0-beta/modules/ml/doc/mldata.html

[23] k-nearest neighbors algorithm – Wikipedia: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[24] Shai Shalev-Shwartz and Shai Ben-David, Understanding Machine Learning: From Theory to Algorithms, 2014, pp. 258-259, ISBN 978-1-107-05713-5.