**An Industry Oriented Major Project Report on**

**“****LICENSE PLATE DETECTION METHODS BASED ON OPENCV”**

Submitted to the

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD**

In partial fulfilment of the requirement for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**BY**

**ANIKET SONI (18WJ1A0521)**

**ROHITH ADDAGATLA (18WJ1A0507)**

**AMMANABOLU GAURAV (18WJ1A0519)**

**Under the Esteemed Guidance of Dr. Ch. Subba Lakshmi,**

**Professor, CSE Dept.**

**GURU NANAK INSTITUTIONS TECHNICAL CAMPUS**

**School of Engineering and Technology Ibrahimpatnam, R.R District- 501506**

**2021-2022**

****

**Department of Computer Science and Engineering**

**CERTIFICATE**

This is to certify that this project report entitled **“LICENSE PLATE DETECTION METHODS BASED ON OPENCV**” by **ANIKET SONI (18WJ1A0521),** **ADDAGATLA ROHITH (18WJ1A0507),** **AMMANABOLU GAURAV (18WJ1A0519)** submitted in partial fulfilment of the requirements for the degree of **Bachelor of Technology** in **Computer Science and Engineering** of the **Jawaharlal Nehru Technological University Hyderabad** during the academic year 2021-2022, is a bonafide record of work carried out under our guidance and supervision.

|  |  |  |
| --- | --- | --- |
| **INTERNAL GUIDE** | **PROJECT CO-ORDINATOR** | **HOD CSE** |
| **DR. CH. SUBBA LAKSHMI** | **MR B SAMIRANA ACHARYA** | **Dr. J RAJESHWAR** |

**EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We wish to express our sincere thanks to **Dr. RISHISAYAL,** Associate Director, GNITC for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We would like to say our sincere thanks to **Dr. J RAJESHWAR, Professor & Head, Department of CSE**, GNITC for providing seamless support and right suggestions are given in the development of the project.

We would like to say our sincere thanks to **Mr B SAMIRANA ACHARYA**, **Asst** **Professor**, Department of CSE, Major Project Coordinator, for providing seamless support and right suggestions are given in the development of the project.

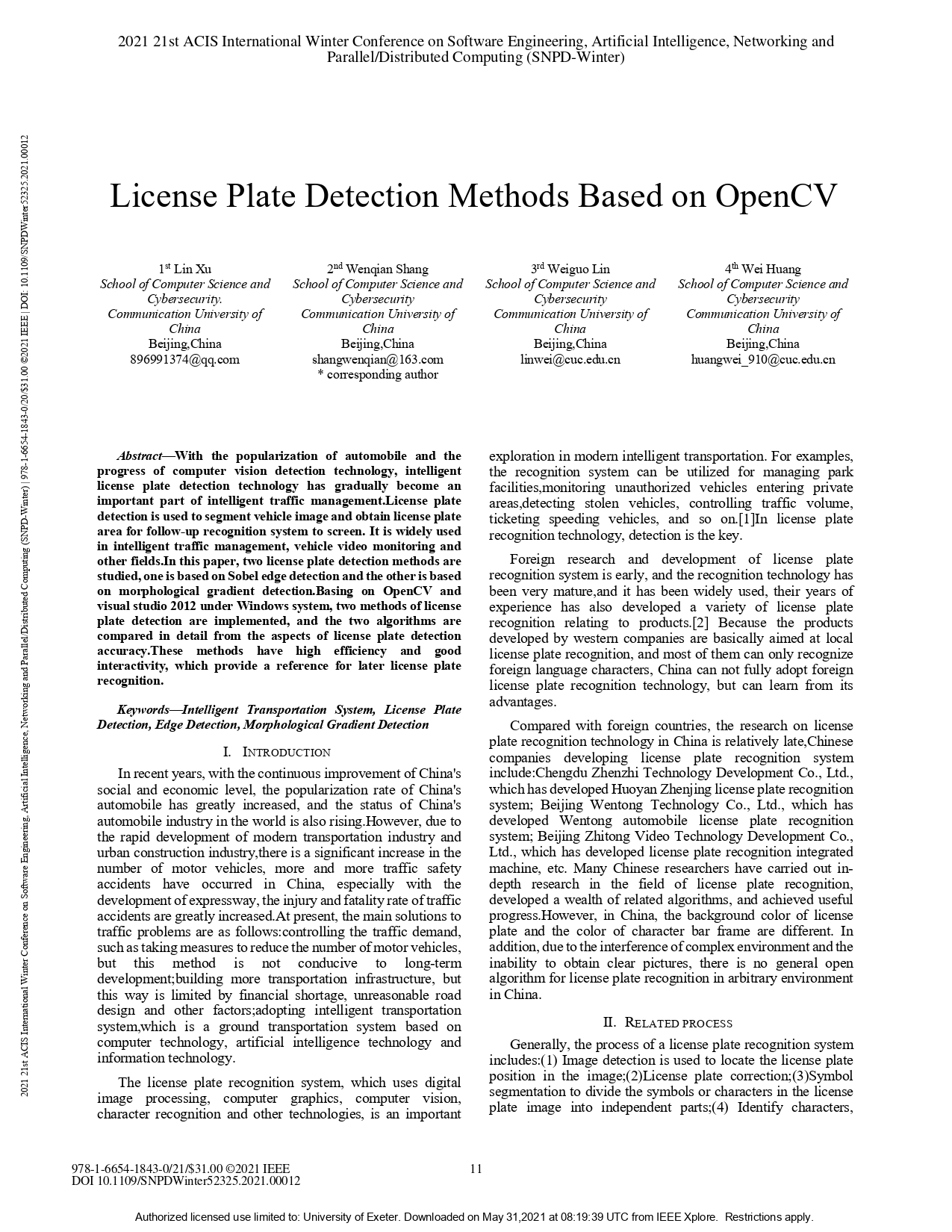
We have been truly blessed to have a wonderful internal guide **DR. CH. SUBBA LAKSHMI,** Professor of CSE, GNITC for guiding us to explore the ramification of our work and we express our sincere gratitude towards him for leading me through the completion of Project.

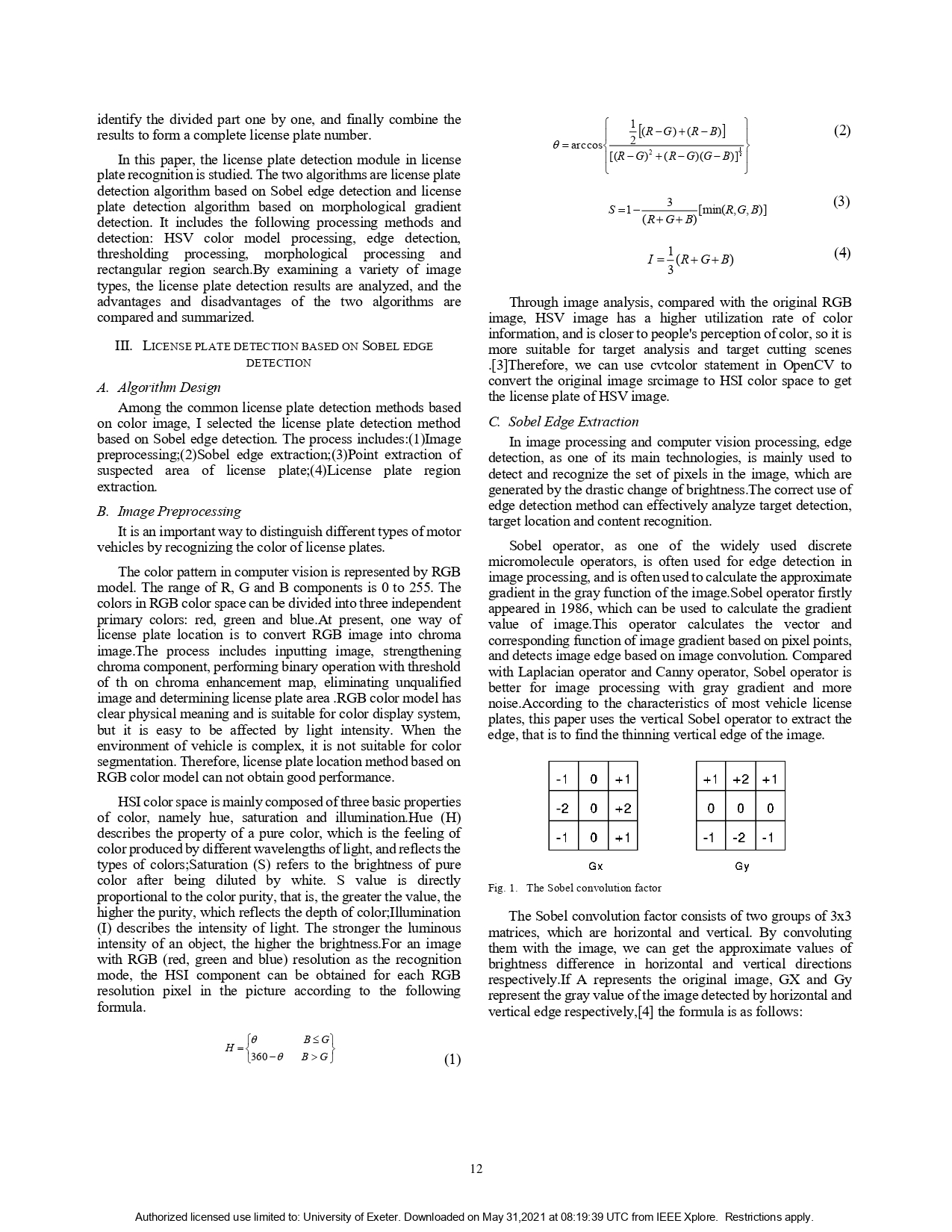
Finally, we would like to thank our family members for their moral support and encouragement to achieve goals.

**ANIKET SONI (18WJ1A0521)**

**ROHITH ADDAGATLA (18WJ1A0507)**

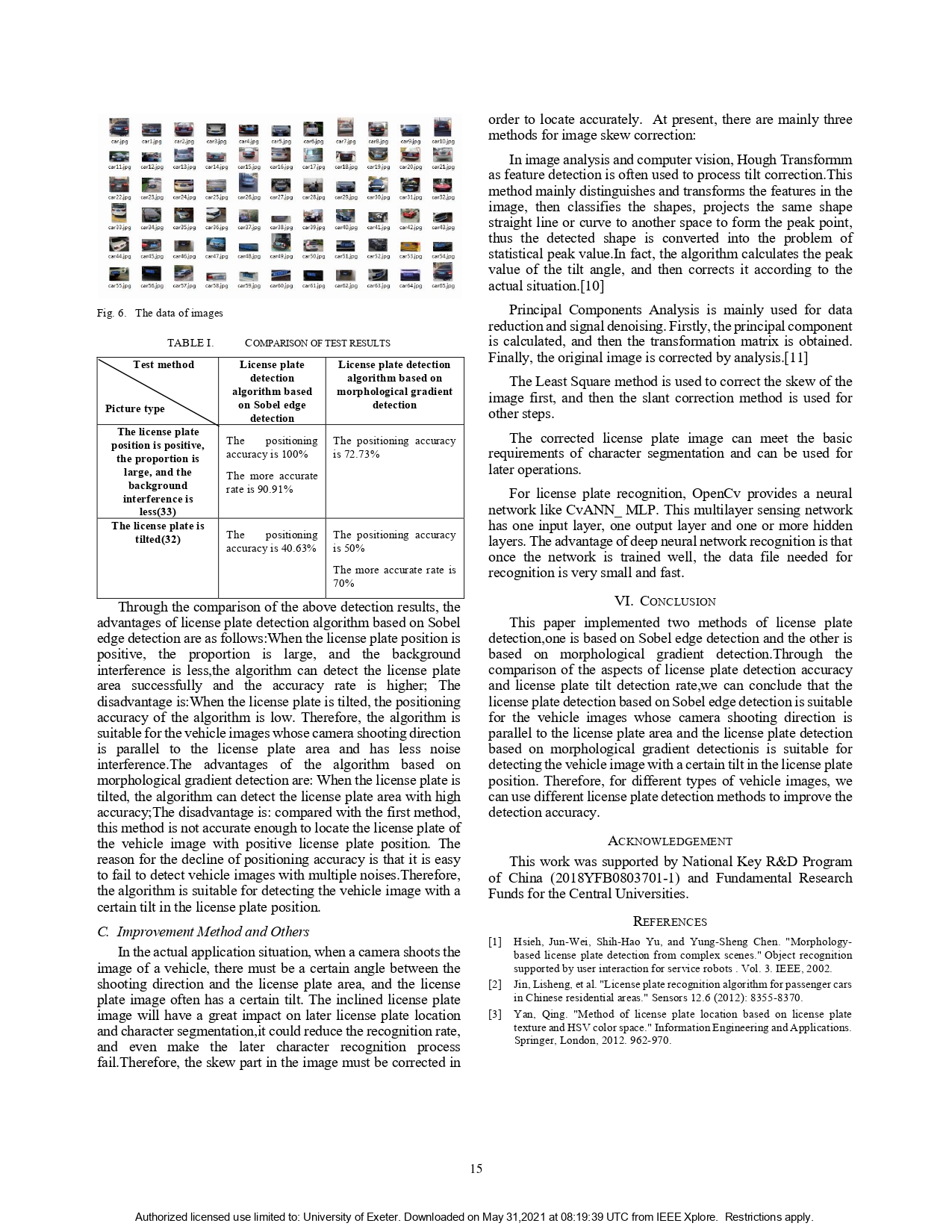
**AMMANABOLU GAURAV (18WJ1A0519)**

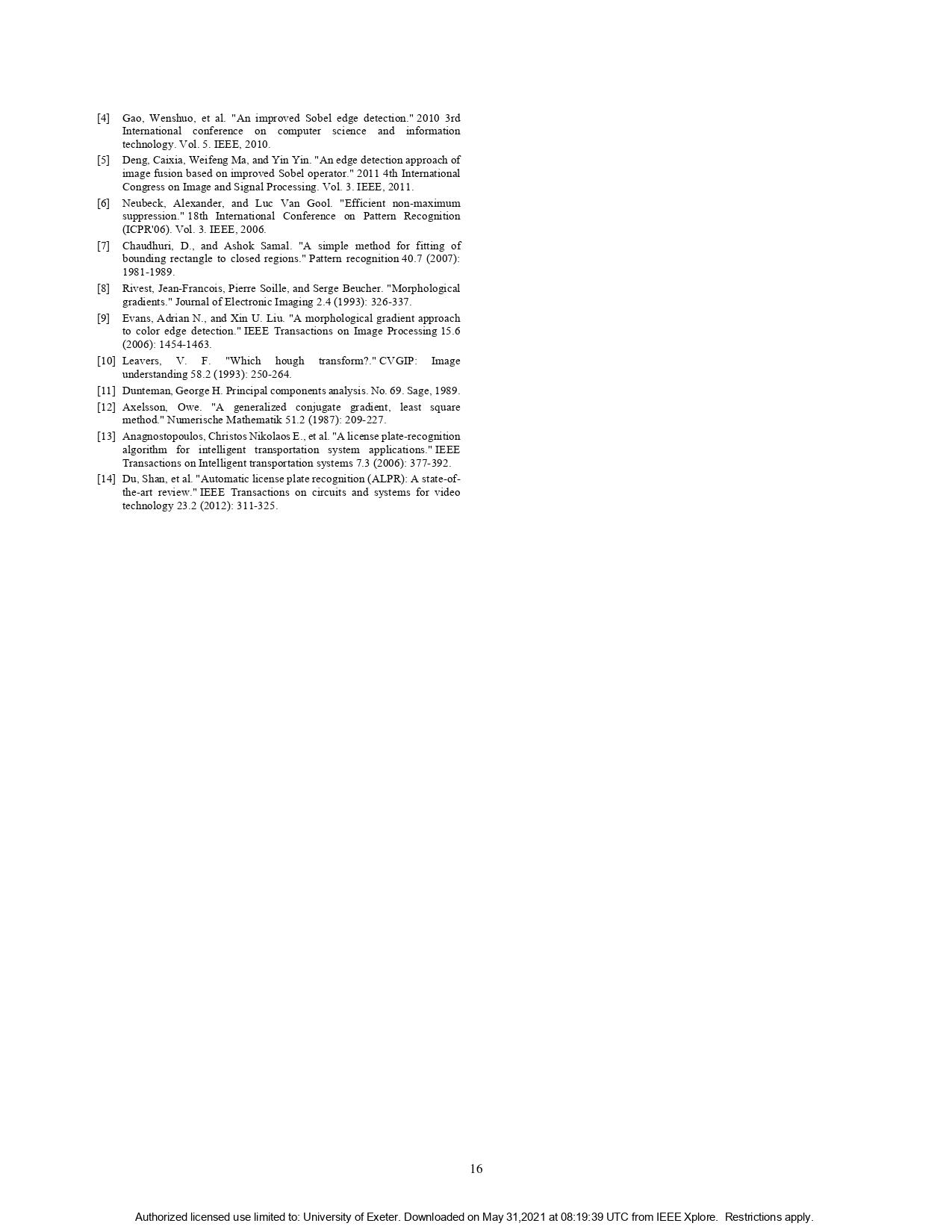
****

****

****

****

****

****

**License Plate Detection Methods Based on OpenCV**

**ABSTRACT**

With the popularization of automobile and the progress of computer vision detection technology, intelligent license plate detection technology has gradually become an important part of intelligent traffic management. License plate detection is used to segment vehicle image and obtain license plate area for follow-up recognition system to screen. It is widely used in intelligent traffic management, vehicle video monitoring and other fields. This work presents a unique enhanced License Plate Detection system using KNN Algorithm. The available strategies square measure susceptible to illumination variance, complicated background and weak-edged license plates and their recognition system fails in it. The proposed new system will sure increase the accuracy and decrease the cost of the recognition in addition of removing the existing system issues. Considering these regards, the proposed system is designed using Knn Algorithm which will be efficient and even robust against noisy data. We prove with the working model and analysis results that the planned model well performs than the prevailing system using Python.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **LIST OF FIGURES**  **LIST OF SYMBOLS** | i  iv  v |
| 1. | **CHAPTER 1 : INTRODUCTION**  1.1 GENERAL  1.2 SCOPE OF THE PROJECT  1.3 OBJECTIVE  1.4 PROBLEM STATEMENT  1.5 EXISTING SYSTEM  1.6 LITERATURE SURVEY  1.7 PROPOSED SYSTEM | 9  9  12  12  13  14  15  16 |
| 2. | **CHAPTER 2 :PROJECT DESCRIPTION**  2.1 GENERAL  2.2 METHODOLOGIES  2.2.1 MODULES NAME  2.2.2 MODULES DESCRIPTION  2.3 TECHNIQUE OR ALGORITHM | 21  21  21  21  22  27 |
| 3. | **CHAPTER 3 : REQUIREMENTS**  3.1 General  3.2 Hardware REQUIREMENTS  3.3 Software REQUIREMENTS  3.4 FUNCTIONAL REQUIREMENTS  3.5 NON-FUNCTIONAL REQUIREMENTS | 28  28  28  29  30  31 |
| 4. | **CHAPTER 4 :SYSTEM DESIGN**  **4.1 general**  **4.2 UML DIAGRAMS**  4.2.1 USE CASE DIAGRAM  4.2.2 CLASS DIAGRAM  4.2.3 OBJECT DIAGRAM  4.2.4 COMPONENT DIAGRAM  4.2.5 DEPLOYMENT DIAGRAM  4.2.6 SEQUENCE DIAGRAM  4.2.7 COLLABORATION DIAGRAM  4.2.8 STATE DIAGRAM  4.2.9 ACTIVITY DIAGRAM  4.2.10 DATA FLOW DIAGRAM  4.2.11 SYSTEM ARCHITECTURE | 32  32  33  33  34  35  36  37  38  39  40  41  42  44 |
| 5. | **CHAPTER 5 : DEVELOPMENT TOOLS**  5.1 PYTHON  5.2 HISTORY OF PYTHON  5.3 IMPORTANCE OF PYTHON  5.4 FEATURES OF PYTHON  5.5 LIBRARIES USED | 45  45  45  45  46  47 |
| 6. | **CHAPTER 6 :IMPLEMENTATION**  6.1 GENERAL  6.2 IMPLEMENTATION | 48  48  48 |

|  |  |  |
| --- | --- | --- |
| 7. | **CHAPTER 7 :SNAPSHOTS**  7.1 GENERAL  7.2 SNAPSHOTS | 51  51  51 |
| 8. | **CHAPTER 8 :SOFTWARE TESTING**  8.1 GENERAL  8.2 DEVELOPING METHODOLOGIES  8.3 TYPES OF TESTING | 56  56  56  56 |
| 9. | **CHAPTER 9 : FUTURE ENHANCEMENT**  9.1 FUTURE ENHANCEMENTS | 59  59 |
| 10 | **CHAPTER 10 :**  10.1 CONCLUSION  10.2 REFERENCES | 60  60  61 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME OF THE FIGURE** | **PAGE NO.** |
| 4.2.1 | Use Case Diagram | 33 |
| 4.2.2 | Class Diagram | 34 |
| 4.2.3 | Object Diagram | 35 |
| 4.2.4 | Component Diagram | 36 |
| 4.2.5 | Deployment Diagram | 37 |
| 4.2.6 | Sequence Diagram | 38 |
| 4.2.7 | Collaboration diagram | 39 |
| 4.2.8 | State Diagram | 40 |
| 4.2.9 | Activity Diagram | 41 |
| 4.2.10 | Data Flow Diagram | 42 |
| 4.2.11 | System Architecture | 44 |

**LIST OF SYSMBOLS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **NOTATION**  **NAME** | **NOTATION** | **DESCRIPTION** |
| 1. | Class | *Class Name*  *-attribute*  *-attribute*  *+operation*  *+operation*  *+operation*  *+ public*  *-private*  *# protected* | Represents a collection of similar entities grouped together. |
| 2. | Association | name  Class B  Class A  Class A  Class B | Associations represents static relationships between classes. Roles represents the way the two classes see each other. |
| 3. | Actor | Class A  Class A  Class B  Class B | It aggregates several classes into a single classes. |
| 4. | Aggregation | Interaction between the system and external environment |

|  |  |  |  |
| --- | --- | --- | --- |
| 5. | Relation  (uses) | uses | Used for additional process communication. |
| 6. | Relation  (extends) | extends | Extends relationship is used when one use case is similar to another use case but does a bit more. |
| 7. | Communication |  | Communication between various use cases. |
| 8. | State | State | State of the processes. |
| 9. | Initial State |  | Initial state of the object |
| 10. | Final state |  | Final state of the object |
| 11. | Control flow |  | Represents various control flow between the states. |
| 12. | Decision box |  | Represents decision making process from a constraint |
| 13. | Use case |  | Interact ion between the system and external environment. |

|  |  |  |  |
| --- | --- | --- | --- |
| 14. | Component |  | Represents physical modules which are a collection of components. |
| 15. | Node |  | Represents physical modules which are a collection of components. |
| 16. | Data Process/State |  | A circle in DFD represents a state or process which has been triggered due to some event or action. |
| 17. | External entity |  | Represents external entities such as keyboard, sensors, etc. |
| 18. | Transition |  | Represents communication that occurs between processes. |
| 19. | Object Lifeline |  | Represents the vertical dimensions that the object communications. |
| 20. | Message | Message | Represents the message exchanged. |

**CHAPTER 1**

**INTRODUCTION**

* 1. **GENERAL**

**Deep Learning:**

Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics. Deep learning discovers intricate structure in large data sets by using the back propagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. Deep convolutional nets have brought about breakthroughs in processing images, video, speech and audio, whereas recurrent nets have shone light on sequential data such as text and speech.

Machine-learning technology powers many aspects of modern society: from web searches to content filtering on social networks to recommendations on e-commerce websites, and it is increasingly present in consumer products such as cameras and smartphones. Machine-learning systems are used to identify objects in images, transcribe speech into text, match news items, posts or products with users’ interests, and select relevant results of search. Increasingly, these applications make use of a class of techniques called deep learning. Conventional machine-learning techniques were limited in their ability to process natural data in their raw form. For decades, constructing a pattern-recognition or machine-learning system required careful engineering and considerable domain expertise to design a feature extractor that transformed the raw data (such as the pixel values of an image) into a suitable internal representation or feature vector from which the learning subsystem, often a classifier, could detect or classify patterns in the input. Representation learning is a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification. Deep-learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level. With the composition of enough such transformations, very complex functions can be learned. For classification tasks, higher layers of representation amplify aspects of the input that are important for discrimination and suppress irrelevant variations. An image, for example, comes in the form of an array of pixel values, and the learned features in the first layer of representation typically represent the presence or absence of edges at particular orientations and locations in the image. The second layer typically detects motifs by spotting particular arrangements of edges, regardless of small variations in the edge positions. The third layer may assemble motifs into larger combinations that correspond to parts of familiar objects, and subsequent layers would detect objects as combinations of these parts. The key aspect of deep learning is that these layers of features are not designed by human engineers: they are learned from data using a general-purpose learning procedure. Deep learning is making major advances in solving problems that have resisted the best attempts of the artificial intelligence community for many years. It has turned out to be very good at discovering intricate structures in high-dimensional data and is therefore applicable to many domains of science, business and government. In addition to beating records in image recognition and speech recognition, it has beaten other machine-learning techniques at predicting the activity of potential drug molecules, analysing particle accelerator data, reconstructing brain circuits, and predicting the effects of mutations in non-coding DNA on gene expression and disease. Perhaps more surprisingly, deep learning has produced extremely promising results for various tasks in natural language understanding, particularly topic classification, sentiment analysis, question answering and language translation. We think that deep learning will have many more successes in the near future because it requires very little engineering by hand, so it can easily take advantage of increases in the amount of available computation and data. New learning algorithms and architectures that are currently being developed for deep neural networks will only accelerate this progress.

In recent years, with the continuous improvement of China's social and economic level, the popularization rate of China's automobile has greatly increased, and the status of China's automobile industry in the world is also rising. However, due to the rapid development of modem transportation industry and urban construction industry, there is a significant increase in the number of motor vehicles, more and more traffic safety accidents have occurred in China, especially with the development of expressway, the injury and fatality rate of traffic accidents are greatly increased. At present, the main solutions to traffic problems are as follows: controlling the traffic demand, such as taking measures to reduce the number of motor vehicles, but this method is not conducive to long-term development; building more transportation infrastructure, but this way is limited by financial shortage, unreasonable road design and other factors; adopting intelligent transportation system, which is a ground transportation system based on computer technology, artificial intelligence technology and information technology.

* 1. **SCOPE OF THE PROJECT**

The aim of the paper is to raise the recognition rate of license plate characters through a combination of three main procedures. All trained and tested characters came from the following two main procedures: extracting license plate, segmenting characters. As a rule, the effectiveness of character processing will affect the effect of selected features, and then further affects the efficacy of chosen classifiers. As expected, experimental results show that our proposed combination of three main procedures does give a very high recognition rate, which can be up to 98.5% for KNN The system has been tested on static snapshots of vehicles, which has divided into several sets according to difficulties.

**1.3 OBJECTIVE**

This work presents a unique enhanced License Plate Detection system using Knn Algorithm. The available strategies square measure susceptible to illumination variance, complicated background and weak-edged license plates and their recognition system fails in it. The proposed new system will sure increase the accuracy and decrease the cost of the recognition in addition of removing the existing system issues. Considering these regards, the proposed system is designed using Knn Algorithm which will be efficient and even robust against noisy data. We prove with the working model and analysis results that the planned model well performs than the prevailing system using Python.

**1.4 PROBLEM STATEMENT**

Usually two cameras are used at a time to increase efficiency. The remote computers then perform further operations like OCR on the stored images sent by the cameras at the lane-level. In order to process the high amount of images stored, a “server farm” is used which comprises of many computers working together. An example of a server farm can be the London Congestion Charge project. The remote computers can be linked with the database which stores the details of the car owners and thus the required information can be obtained. Using this information the culprit can be caught. License plate detection is used to segment vehicle image and obtain license plate area for follow-up recognition system to screen. It is widely used in intelligent traffic management, vehicle video monitoring and other fields. We first introduce how to locate license plates and extract their corresponding regions, then segment these characters on located license plate, and finally use K-nearest neighbor (KNN) classifiers to recognize these segmented characters. We prove with the working model and analysis results that the planned model well performs than the prevailing system using Python.

**1.5 EXISTING SYSTEM**

* Existing ANPR System using OCR, the hub of the system is the OCR (Optical Character Recognition system) which is used to extract the alphanumeric characters present on the number plate. To do this it first uses a series of image manipulation techniques to detect, normalize and enhance the image of the number plate. There are two components in the system, the cameras at the front-end and the remote computers at the back-end.
* Usually two cameras are used at a time to increase efficiency. The remote computers then perform further operations like OCR on the stored images sent by the cameras at the lane-level. In order to process the high amount of images stored, a “server farm” is used which comprises of many computers working together. An example of a server farm can be the London Congestion Charge project. The remote computers can be linked with the database which stores the details of the car owners and thus the required information can be obtained. Using this information the culprit can be caught.

**1.5.1 EXISTING SYSTEM DISADVANTAGES**

* In case the number is read partially, the remote computer might identify the number plate incorrectly or would not be able to decrypt at all.
* Hazy images can also make the detection process erroneous or there is a possibility of no detection at all.
* Angular detection is not possible in case of ANPR as the rectangulation algorithm, implemented in OCR is not possible thus characters may be misread/ overlapped.

**1.6 LITERATURE SURVEY**

**TITLE :** Morphology-based license plate detection from complex scenes

**AUTHORS :** Jun-Wei Hsieh; Shih-Hao Yu; Yung-Sheng Chen

**YEAR :** 2012

**DESCRIPTION**

This paper presents a morphology-based method for detecting license plates from cluttered images. The proposed system consists of three major components. At the first, a morphology-based method is proposed to extract important contrast features as guides to search the desired license plates. The contrast feature is robust to lighting changes and invariant to several transformations like scaling, translation, and skewing. Then, a recovery algorithm is applied for reconstructing a license plate if the plate is fragmented into several parts. The last step is to do license plate verification. The morphology based method can significantly reduce the number of candidates extracted from the cluttered images and thus speeds up the subsequent plate recognition. Under the experimental database, 128 examples got from 130 images were successfully detected. The average accuracy of license plate detection is 98%. Experimental results show that the proposed method improves the state-of-the-art work in terms of effectiveness and robustness of license plate detection.

**TITLE :** Method of license plate location based on license plate texture and HSV color space

**AUTHORS :** Huili Han, Runping Han

**YEAR :** 2012

**DESCRIPTION**

A new method of license plate location is designed in the paper and it is divided into two steps, which are rough location step based on gray image processing and precise location step based on color image processing. Firstly, the color car image is converted to the gray one from which horizontal and vertical edge images are extracted using Haar wavelet and Gabor filter. The rough location is achieved based on the specific projection features of these edge images. Then the color plate image roughly located by the first step is converted to HSI space. The binary image is obtained through the threshold segmentation method according to the color information of license plate. The plate's precise position parameters are determined through making projection analysis of the binary image. The simulation result shows that the location method in this paper is efficient and effective.

**TITLE :** Automatic license plate recognition (ALPR): A state-of- the-art review

**AUTHORS :** Shan Du; Mahmoud Ibrahim; Mohamed Shehata; Wael Badawy

**YEAR :** 2012

**DESCRIPTION**

Automatic license plate recognition (ALPR) is the extraction of vehicle license plate information from an image or a sequence of images. The extracted information can be used with or without a database in many applications, such as electronic payment systems (toll payment, parking fee payment), and freeway and arterial monitoring systems for traffic surveillance. The ALPR uses either a color, black and white, or infrared camera to take images. The quality of the acquired images is a major factor in the success of the ALPR. ALPR as a real-life application has to quickly and successfully process license plates under different environmental conditions, such as indoors, outdoors, day or night time. It should also be generalized to process license plates from different nations, provinces, or states. These plates usually contain different colors, are written in different languages, and use different fonts; some plates may have a single color background and others have background images. The license plates can be partially occluded by dirt, lighting, and towing accessories on the car. In this paper, we present a comprehensive review of the state-of-the-art techniques for ALPR. We categorize different ALPR techniques according to the features they used for each stage, and compare them in terms of pros, cons, recognition accuracy, and processing speed. Future forecasts of ALPR are given at the end.

**TITLE :** An edge detection approach of image fusion based on improved Sobel operator

**AUTHORS :** Caixia Deng; Weifeng Ma; Yin Yin

**YEAR :** 2011

**DESCRIPTION**

Anti-noise ability and edge continuity of Sobel edge detection algorithm are poor. In order to solve these problems, an improved method of Sobel operator is given in this paper. In addition, making use of fusion technology, a kind of method combined with improved Sobel operator, wavelet transform, Canny algorithm and Prewitt operator is put forward, which keeps their respective advantages. Experiments show that the fusion image effectively improves the accuracy of edge detection and gets a quite ideal edge detection effect.

**TITLE :** An improved Sobel edge detection

**AUTHORS :** Wenshuo Gao; Xiaoguang Zhang; Lei Yang; Huizhong Liu

**YEAR :** 2010

**DESCRIPTION**

This paper proposes a method which combines Sobel edge detection operator and soft-threshold wavelet de-noising to do edge detection on images which include White Gaussian noises. In recent years, a lot of edge detection methods are proposed. The commonly used methods which combine mean de-noising and Sobel operator or median filtering and Sobel operator can not remove salt and pepper noise very well. In this paper, we firstly use soft-threshold wavelet to remove noise, then use Sobel edge detection operator to do edge detection on the image. This method is mainly used on the images which includes White Gaussian noises. Through the pictures obtained by the experiment, we can see very clearly that, compared to the traditional edge detection methods, the method proposed in this paper has a more obvious effect on edge detection.

**1.7 PROPOSED SYSTEM**

* We first introduce how to locate license plates and extract their corresponding regions, then segment these characters on located license plate, and finally use K-nearest neighbor (KNN) classifiers to recognize these segmented characters.
* The K-nearest-neighbor (KNN) algorithm measures the distance between a query scenario and a set of scenarios in the data set. KNN is more appropriate than existing methods and its recognition rate is up to 98.51 % on average.

**1.7.1 Proposed System Advantages**

* No Training Period, In other words, there is no training period for it. It stores the training dataset and learns from it only at the time of making real time predictions. This makes the KNN algorithm much faster than other algorithms that require training e.g. SVM, Linear Regression etc.
* Since the KNN algorithm requires no training before making predictions, new data can be added seamlessly which will not impact the accuracy of the algorithm.
* KNN is very easy to implement. There are only two parameters required to implement KNN i.e. the value of K and the distance function (e.g. Euclidean or Manhattan etc.)

**CHAPTER 2**

**PROJECT DESCRIPTION**

**2.1 GENERAL:**

Because of this, we often detect several candidates for the plate by different algorithms. There are several heuristics, which are used to determine the cost of selected candidates according to their properties. These heuristics have been chosen on ad-hoc basis during the practical experimentations. This recognition logic sorts candidates according to their cost from the most suitable to the least suitable. Then, the most suitable candidate is examined by a deeper heuristic analysis. The deeper analysis definitely accepts, or rejects the candidate. As there is a need to analyze individual characters, this type of analysis consumes big amount of processor time.

* 1. **METHODOLOGIES**

**2.2.1 MODULES NAME:**

**This project having the following 9 modules:**

* **Capture Image**
* **Binarization**
* **Edge Detection**
* **Number Plate Localization**
* **Deskewing**
* **Segmentation**
* **Feature Extraction**
* **Character Recognition**
* **Digital Output**

**2.2.2 MODULE DESCRIPTION:**

**CAPTURE IMAGE:**

In image acquisition where vehicle images are acquire using the camera, Image can be input to the system by different methods by analog camera or by digital camera, nowadays digital technology has their advantages so better input method is by digital cameras or by direct digital photos. The camera senses the vehicle moving on the road and immediately captures either front of rare view of vehicle depending on the position of the vehicle.

**BINARIZATION:**

The captured image is then binaries, binarization is a widely used technique, its process is to first determine a gray threshold (GT) according to some objective criteria and then assigns each pixel (Ixy) to one class, windowing on the grey threshold image is then taken place, different methods are used for binarization, binarization process is broadly classified as global or local depending on how the threshold is calculated. In this we propose a new edge based adaptive thresholding method which is capable of preserving the text region of an image robustly in real life conditions. This method uses local edge properties in a window to compute threshold which can be effectively used for text region extraction from the vehicle license plate images.

**EDGE DETECTION:**

After binarizing the image next step is to detect the edge of the license plate from the body of the vehicle. Let an image foreground be represented by G1 and background by G2 gray levels and let G1 > G2. Typically the gray levels between background and foreground change gradually. For binarizing such an image we should have a threshold GT such that G1≥GT ≥G2. A typical edge detection algorithm would mark edges at all or some of the locations marked by arrows in Figure 2. We exploit the edge image and use the edge pixels to identify the threshold (GT ) for binarization. A computationally simple way to compute GT is to take the average gray value of the all the pixels on the edges.

**NUMBER PLATE LOCALIZATION:**

After detecting the edges of the vehicle next step is to find out where the number plate is located on the body of the vehicle, due to different size and shape of vehicles it is important to detect the location of the license plate area a mong the body of the vehicle, Let us consider the number plate as a “rectangular area with increased occurrence of horizontal and vertical edges”. This process can sometimes detect a wrong area that does not correspond to a number plate. Because of this, we often detect several candidates for the plate by different algorithms. There are several heuristics, which are used to determine the cost of selected candidates according to their properties. These heuristics have been chosen on ad-hoc basis during the practical experimentations. This recognition logic sorts candidates according to their cost from the most suitable to the least suitable. Then, the most suitable candidate is examined by a deeper heuristic analysis. The deeper analysis definitely accepts, or rejects the candidate. As there is a need to analyze individual characters, this type of analysis consumes big amount of processor time.

**DESKEWING:**

The captured rectangle plate 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 plates. The number plate 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. Hough transform is special operation, which is used to extract features of specific shape within a picture. The classical Hough transform is used for the detection of lines. The Hough transform is widely used for miscellaneous purpose in the problematic of machine vision, but here it is used to detect the skew of capture plate, and also to compute an angle of skew localization. After capturing the front or rear view of the vehicle, the first step is to detect the exact area of the number plate from the captured image. Let us define the number plate as a “rectangular area with increased occurrence of horizontal and vertical edges”. The high density of horizontal and vertical edges on a small area is in many cases caused by contrast characters of a number plate, but not in every case. This process can sometimes detect a wrong area that does not correspond to a number plate. Because of this, we often detect several candidates for the plate by different algorithms. There are several heuristics, which are used to determine the cost of selected candidates according to their properties. These heuristics have been chosen on ad-hoc basis during the practical experimentations. The recognition logic sorts candidates according to their cost from the most suitable to the least suitable. Then, the most suitable candidate is examined by a deeper heuristic analysis. The deeper analysis definitely accepts, or rejects the candidate. As there is a need to analyze individual characters, this type of analysis consumes big amount of processor time.

**SEGMENTATION:**

By this step characters on license plate are segmented and identify. This step is the most important step in license plate recognition because all further steps rely on it. There are many factors that cause the character segmentation task difficult, such as image noise, plate frame, rivet, space mark, plate rotation and illumination variance. We here propose the algorithm that is quite robust and gives significantly good results on images having the above mentioned problems. for the segmentation pre-processing is required by conversion to gray scale and binarization. We use the method of horizontal projection for segmentation, If we assume only one row plate, the segmentation is the process of finding horizontal boundaries between characters. The segmented area of the plate can contain redundant space and other undesirable elements besides the characters. Since the “segment” has been processed by an adaptive thresholding filter, it contains only black and white pixels. The neighboring pixels are grouped together into larger pieces, and one of them is a character. Our goal is to divide the segment into several pieces, and identify only one piece representing the regular character. The second phase of the segmentation is an enhancement of segments. The piece chosen by the heuristic is then converted to a monochrome bitmap image. Each such image corresponds to one horizontal segment. These images are considered as an output of the segmentation phase. Normalization of Character The first step in character normalization is to adjust the brightness and contrast of the segmented image, after adjusting brightness and contrast the second step is resize the characters in uniform dimension, the third step is feature extraction, the feature extraction algorithm extracts appropriate descriptors from the normalized characters, the normalization of the character is important step because brightness and contrast characteristics of segmented characters are varying due to different light conditions during the image is captured, techniques of global and adaptive thresholding are used to obtain monochrome representations of processed character segments. The monochrome (black & white) representation of image is more appropriate for analysis, because it defines clear boundaries of contained character.

**FEATURE EXTRACTION**:

Before extracting feature descriptors from a bitmap representation of a character, it is necessary to normalize it into unified dimensions. We define the term “re-sampling” as the process of changing dimensions of the character. As original dimensions of un-normalized characters are usually higher than the normalized ones, the characters are in most cases down sampled. When we say down sample, we reduce information contained in the processed image. There are several methods of re-sampling, such as the pixel re-size, bilinear interpolation, or the weightedaverage re-sampling. We cannot determine which method is the best in general, because the successfulness of a particular method depends on many factors. To recognize a character from a bitmap representation, there is a need to extract feature descriptor of such bitmap. As the extraction method significantly affects the quality of whole OCR process, it is very important to extract features which will be invariant towards the various light conditions, used font type and deformation of characters caused by a skew of the image. The description of normalized character is based on its external characteristics because we deal only with properties such as character shape. Then, the vector of descriptors includes characteristics such as number of lines, bays, lakes, the amount of horizontal, vertical, or diagonal edges etc. The feature extraction is a process of transformation of data from a bitmap representation into a form of descriptors, which are more suitable for computers. If we associate similar instances of the same character into the classes, then the descriptors of characters from the same class should be geometrically closed to each other in the vector space. This is the basic assumption for successfulness of the pattern recognition process.

**CHARACTER RECOGNITION**:

By number plate extraction step final result is founded. Consider following figure as an final extracted license plates. The variations of the plate types or environments cause challenges in the detection and recognition of license plates. They are summarized as follows: 1) Location: Plates exist in different locations of an image. 2) Quantity: An image may contain no or many plates. 3) Size: Plates may have different sizes due to the camera distance and the zoom factor. 4) Colour: Plates may have various characters and background colours due to different plate types or capturing devices. 5) Font: Plates of different nations may be written in different fonts and language. In this paper, the feature is referred to as blockbinary- pixel-sum, which was applied in the paper. They provided an automatic license plate.

**DIGITAL OUTPUT**:

After Character Recognition process the recognized no apply to Python software for further processor using KNN algorithm, the Python Software provide number of license plate at the output

**2.3 TECHNIQUE USED OR ALGORITHM USED**

**NUMBER PLATE LOCALIZATION:**

After detecting the edges of the vehicle next step is to find out where the number plate is located on the body of the vehicle, due to different size and shape of vehicles it is important to detect the location of the license plate area a mong the body of the vehicle, Let us consider the number plate as a “rectangular area with increased occurrence of horizontal and vertical edges”. This process can sometimes detect a wrong area that does not correspond to a number plate. Because of this, we often detect several candidates for the plate by different algorithms. There are several heuristics, which are used to determine the cost of selected candidates according to their properties. These heuristics have been chosen on ad-hoc basis during the practical experimentations. This recognition logic sorts candidates according to their cost from the most suitable to the least suitable. Then, the most suitable candidate is examined by a deeper heuristic analysis. The deeper analysis definitely accepts, or rejects the candidate. As there is a need to analyze individual characters, this type of analysis consumes big amount of processor time.**CHAPTER 3**

**REQUIREMENTS ENGINEERING**

**3.1 GENERAL**

These are the requirements for doing the project. Without using these tools and software’s we can’t do the project. So we have two requirements to do the project. They are

1. Hardware Requirements.

2. Software Requirements.

**3.2 HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shouls what the system do and not how it should be implemented.

* PROCESSOR : DUAL CORE 2 DUOS.
* RAM : 4GB DD RAM
* HARD DISK : 250 GB

**3.3 SOFTWARE REQUIREMENTS**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team’s progress throughout the development activity.

* OPERATING SYSTEM : WINDOWS 7/8/10
* PLATFORM : SPYDER3
* PROGRAMMING LANGUAGE : PYTHON, HTML
* FRONT END : SPYDER3

**3.4 FUNCTIONAL REQUIREMENTS**

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behaviour, and outputs. We sequentially explored different hyperparameter value combinations from the given possible value ranges and tuned them until we obtained a state-of-the-art accuracy of at least 96%. We also analyzed the performance of different ML models in terms of accuracy, precision, recall, and F1-score, and found out that SVM produced the most accurate result

**3.5 NON-FUNCTIONAL REQUIREMENTS**

The major non-functional Requirements of the system are as follows

* **Usability**

The system is designed with completely automated process hence there is no or less user intervention.

* **Reliability**

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

* **Performance**

This system is developing in the high-level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

* **Supportability**

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having Python, built into the system.

* **Implementation**

The system is implemented in web environment using struts framework. The apache tomcat is used as the web server and windows xp professional is used as the platform. Interface the user interface is based on Struts provides HTML Tag.

**CHAPTER 4**

**DESIGN ENGINEERING**

**4.1 GENERAL**

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

**4.2 UML DIAGRAMS**

**4.2.1 USE CASE DIAGRAM**



**EXPLANATION:**

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

**4.2.2 CLASS DIAGRAM:**



**EXPLANATION**

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

**4.2.3 OBJECT DIAGRAM:**



**EXPLANATION:**

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

**4.2.4 COMPONENT DIAGRAM**



**EXPLANATION**

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

**4.2.5 DEPLOYMENT DIAGRAM:**



**EXPLANATION:**

Deployment diagrams is a kind of structure diagram used in the user has a link with a user dataset it was also have a pre-processing a data and it has a training and testing data from the dataset. It has a predictions to predict a values and it has a detect a number plate detection.

**4.2.6 SEQUENCE DIAGRAM:**



**EXPLANATION:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

**4.2.7 COLLABORATION DIAGRAM:**



**EXPLANATION:**

A collaboration diagram, also called a communication diagram user has a login with a data. Dataset has it detects a fake news data. Classifier it has a apply algorithms in a KNN, decision trees and links to the predictions. Pre-processing it also have a removes a unwanted data from predictions to predict values.

**4.2.8 STATE DIAGRAM:**



**EXPLANATION:**

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

**4.2.9 ACTIVITY DIAGRAM:**



**EXPLANATION:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

**4.2.10 DATA FLOW DIAGRAM**

**Level 0:**

Video

Data Acquisition

Number Plate Detection

Pre-Processing

Segmentation, Extraction

**Level 1:**

Binarization, Noise Removal

Video

Acquisition

Number Plates

Frames, Width, Height, Convert to Gray

Character Segmentation, Extraction, Number Plate

Pre-Processing

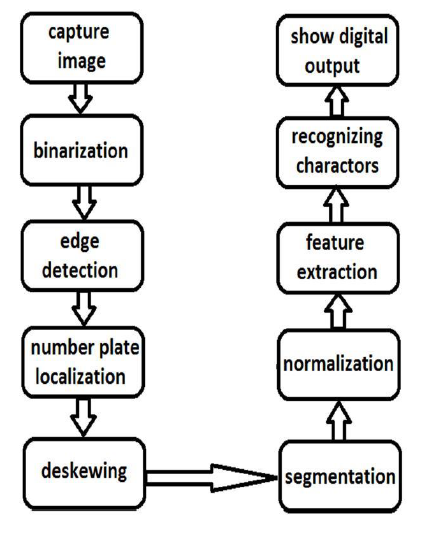
Detection

**EXPLANATION:**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

**4.2.11 SYSTEM ARCHITECTURE**



**CHAPTER 5**

**DEVELOPMENT TOOLS**

**5.1 Python**

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

**5.2 History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

**5.3 Importance of Python**

* **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**5.4 Features of Python**

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* IT supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**5.5 Libraries used in python:**

* numpy - mainly useful for its N-dimensional array objects.
* pandas - Python data analysis library, including structures such as dataframes.
* matplotlib - 2D plotting library producing publication quality figures.
* scikit-learn - the machine learning algorithms used for data analysis and data mining tasks.



Figure : NumPy, Pandas, Matplotlib, Scikit-learn

**CHAPTER 6**

**IMPLEMENTATION**

**6.1 GENERAL**

After training the model, it is saved and then the saved model is used for prediction. A full stack web application with NodeJs and Express Handlebars has been developed using this model for prediction. It incorporates different logics to make it a product which can be used with certain improvements in place. Depicts the Flow The suggested system is depicted in this diagram.

The CNN model is applied in the first part wherein the input is an image. After being processed, one of the classes out of the 43 classes is obtained as the output. If a certain image is not containing a traffic sign, then the user gets a prompt of “No Sign Detected”. This is done by analysing the output array of the "model.predict" function in python. The "model.predict" function returns an array of values representing how closely the image falls under each of the 43 classes and finally predicts the class based on the highest value.

**6.2 IMPLEMENTATION:**

**app.py**

1. from flask import \*
2. import os
3. from werkzeug.utils import secure\_filename
4. from keras.models import load\_model
5. import numpy as np
6. from PIL import Image
7. import pyttsx3
8. app = Flask(\_\_name\_\_)
9. # Classes of trafic signs
10. classes = { 0:'Speed limit (20km/h)',
11. 1:'Speed limit (30km/h)',
12. 2:'Speed limit (50km/h)',
13. 3:'Speed limit (60km/h)',
14. 4:'Speed limit (70km/h)',
15. 5:'Speed limit (80km/h)',
16. 6:'End of speed limit (80km/h)',
17. 7:'Speed limit (100km/h)',
18. 8:'Speed limit (120km/h)',
19. 9:'No passing',
20. 10:'No passing veh over 3.5 tons',
21. 11:'Right-of-way at intersection',
22. 12:'Priority road',
23. 13:'Yield',
24. 14:'Stop',
25. 15:'No vehicles',
26. 16:'Vehicle > 3.5 tons prohibited',
27. 17:'No entry',
28. 18:'General caution',
29. 19:'Dangerous curve left',
30. 20:'Dangerous curve right',
31. 21:'Double curve',
32. 22:'Bumpy road',
33. 23:'Slippery road',
34. 24:'Road narrows on the right',
35. 25:'Road work',
36. 26:'Traffic signals',
37. 27:'Pedestrians',
38. 28:'Children crossing',
39. 29:'Bicycles crossing',
40. 30:'Beware of ice/snow',
41. 31:'Wild animals crossing',
42. 32:'End speed + passing limits',
43. 33:'Turn right ahead',
44. 34:'Turn left ahead',
45. 35:'Ahead only',
46. 36:'Go straight or right',
47. 37:'Go straight or left',
48. 38:'Keep right',
49. 39:'Keep left',
50. 40:'Roundabout mandatory',
51. 41:'End of no passing',
52. 42:'End no passing vehicle > 3.5 tons',
53. 43:'No Sign Detected'
54. }def image\_processing(img):
55. model = load\_model('./model/traffic.h5')
56. data=[]
57. image = Image.open(img)
58. image = image.resize((30,30))
59. data.append(np.array(image))
60. X\_test=np.array(data)
61. Y\_pred = model.predict\_classes(X\_test)
62. return Y\_pred
63. @app.route('/')
64. @app.route('/index')
65. def index(): return render\_template('index.html')
67. @app.route('/login')
68. def login():
69. return render\_template("login.html")
70. @app.route('/first', methods=['GET'])
71. def first():
72. # Main page
73. return render\_template('first.html')
74. @app.route('/predict', methods=['GET', 'POST'])
75. def upload():
76. if request.method == 'POST':
77. # Get the file from post request
78. f = request.files['file']
79. file\_path = secure\_filename(f.filename)
80. f.save(file\_path)
81. # Make prediction
82. result = image\_processing(file\_path)
83. s = [str(i) for i in result]
84. a = int("".join(s))
85. result = "Predicted Traffic Sign is: " +classes[a]
86. os.remove(file\_path)
87. engineio = pyttsx3.init()
88. engineio.say(result)
89. results=engineio.runAndWait()
90. return result
91. return results
92. return None
93. @app.route('/performance')
94. def performance():
95. return render\_template("performance.html")
96. @app.route('/chart')
97. def chart():
98. return render\_template("chart.html")
99. if \_\_name\_\_ == '\_\_main\_\_':
100. app.run(port=5000,debug=True)

**CHAPTER 7**

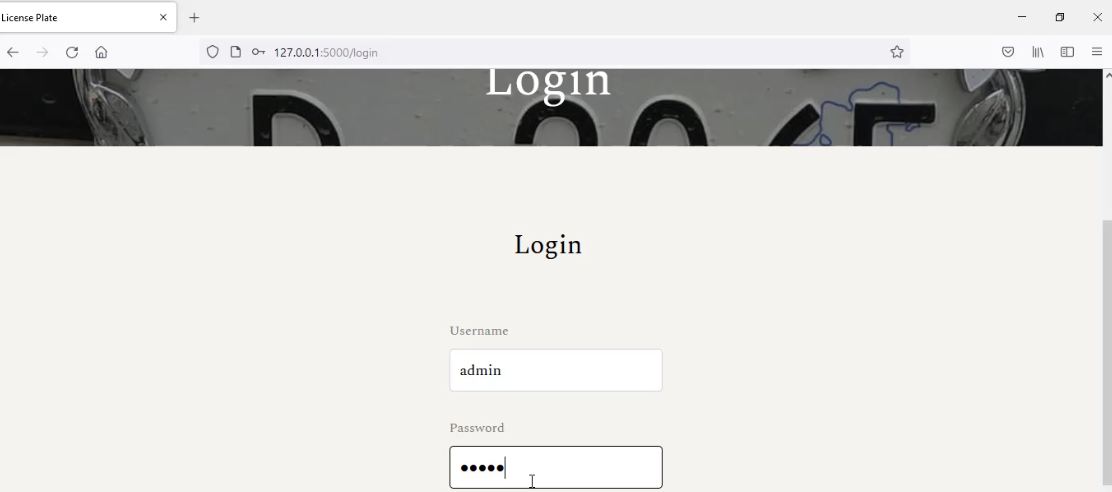
**SNAPSHOTS**

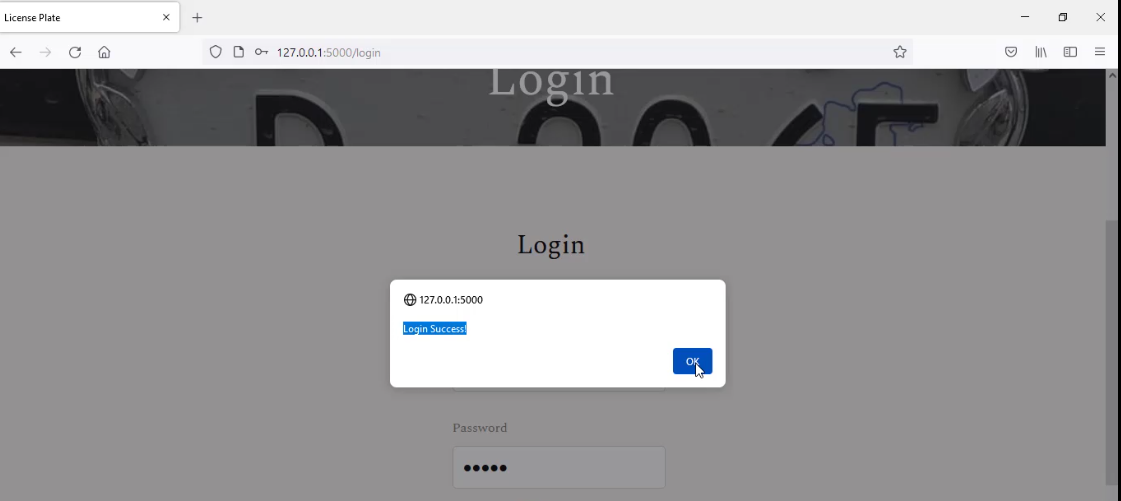
**7.1 GENERAL**

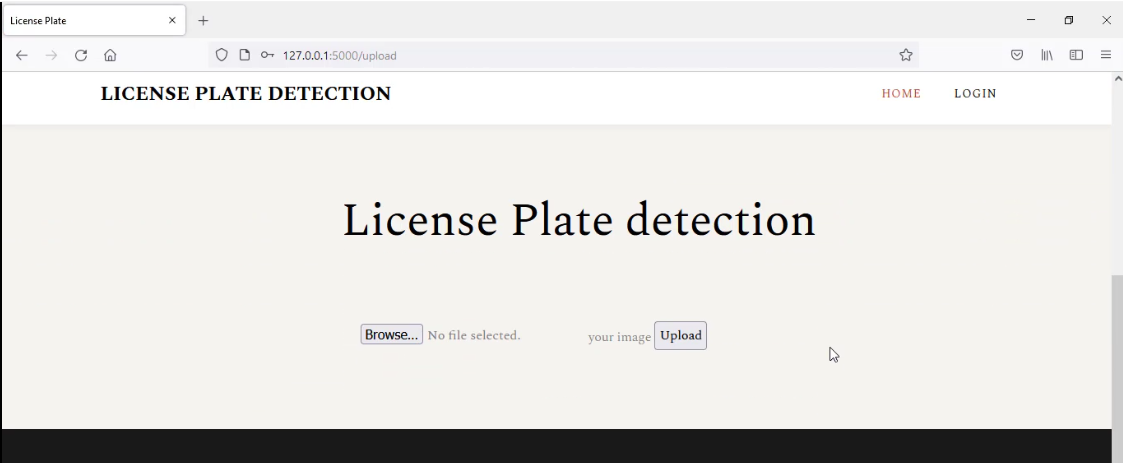
This project is implements like application using python and the Server process is maintained using the SOCKET & SERVERSOCKET and the Design part is played by Cascading Style Sheet.

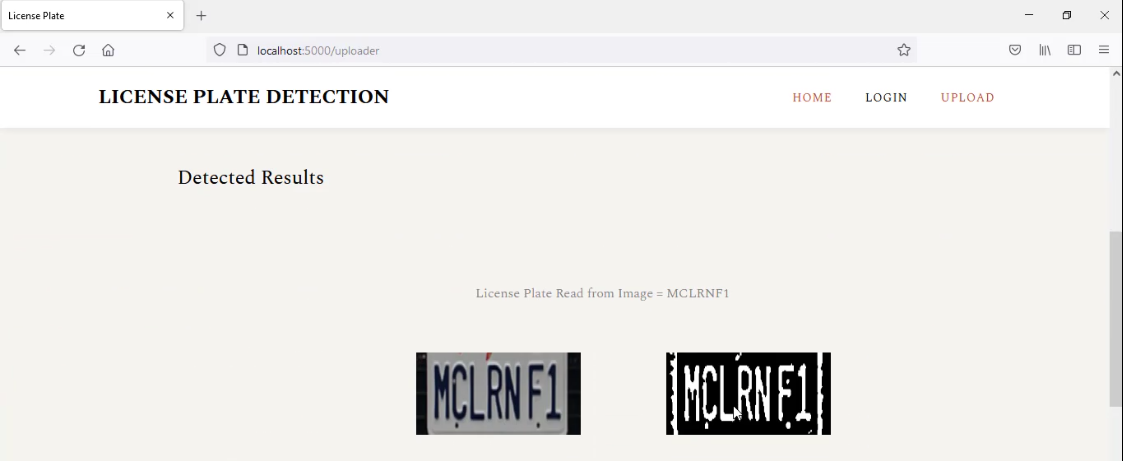
**7.2 SNAPSHOTS**











**CHAPTER 8**

**SOFTWARE TESTING**

**8.1 GENERAL**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**8.2 DEVELOPING METHODOLOGIES**

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

**8.3Types of Tests**

**8.3.1 Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**8.3.2 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

**8.3.3 System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**8.3.4 Performance Test**

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

**8.3.5 Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

**8.3.6 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Acceptance testing for Data Synchronization:**

* The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
* The Route add operation is done only when there is a Route request in need
* The Status of Nodes information is done automatically in the Cache Updation process

**8.2.7 Build the test plan**

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

**CHAPTER 9**

**FUTURE ENHANCEMENT**

**9.1 FUTURE ENHANCEMENT**

In future, currently there are certain restrictions on parameters like speed of the vehicle, script on the number plate, cleanliness of number plate, quality of captured image, skew in the image which can be aptly removed by enhancing the algorithm further.

**CHAPTER 10**

**CONCLUSION & REFERENCE**

**10.1 CONCLUSION**

The aim of the paper is to raise the recognition rate of license plate characters through a combination of three main procedures. All trained and tested characters came from the following two main procedures: extracting license plate, segmenting characters. As a rule, the effectiveness of character processing will affect the effect of selected features, and then further affects the efficacy of chosen classifiers. As expected, experimental results show that our proposed combination of three main procedures does give a very high recognition rate, which can be up to 98.5% for KNN The system has been tested on static snapshots of vehicles, which has divided into several sets according to difficulties. Sets of blurry and skewed snapshots give worse recognition rates than a set of snapshots, which has been captured clearly. The objective of the tests was not to find 100% recognizable set of snapshots, but to test the invariance of the algorithms on random snapshots systematically classified to the sets according to their properties.

**10.2 REFERENCE**

[1] Su-Hyun Lee,Young- Soo Seok and Eung-Joo Lee Department of information/Communication Eng., TongMyong Univ.of Information Technology: Multi-National Integrated Car-License Plate Recognition System Using Geometrical Feature and Hybrid Pattern Vector.

[2] V. Franc and V. Hlaváč, Statistical Pattern Recognition Toolbox for MATLAB: User’s Guide, Center for Machine Perception, Czech Technical University, 2004.

[3] V. Wu, R. Manmatha, and E.M. Riseman. Automatic text detection and recognition. Proceedings of Image Understanding Workshop, pages 707–712, 1997.

[4] V.L Lajish and Sunil Kopparapu. Mobile phone based vehicle license plate recognition for road policing. Experience Workshop, New Delhi, December 18, 2009 Co-located and Concurrently with Third International Conference on Pattern Recognition and Machine Intelligence(PReMI’09), 2009.

[5] W. Zhou, H. Li, Y. Lu, and Q. Tian, “Principal visual word discovery for automatic license plate detection,” IEEE Trans. Image Process., vol. 21, no. 9, pp. 4269-4279, 2012.

[6] Y. Hasan and L. Karam, “Morphological Text Extraction from Images,” IEEE Trans. Image Process., vol. 9, no. 11, pp.1978-1983, 2000.

[7] Z.-X. Chen, C.-Y. Liu, and F.-L. Chang, “Automatic license-plate location and recognition based on feature salience,” IEEE Trans. Veh. Technol., vol. 58, no. 7, pp. 3781-3785, 2009.

[8] L. Angeline, M. Y. Choong, F. Wong, and K. T. K. Teo, “Tracking and localisation of moving vehicle license plate via signature analysis,” Int. Conf. Mechatronics: Integr. Eng. Ind. Soc. Dev., 2011.

[9] Luis Salgado, Jose M. Mene’ndex, Enrique Rendnn and Narciso Garcia (1999): Automatic Car Plate Detection and Recognition through Intelligent Vision Engineering, IEEE.

[10] Muhammad Tahir Qadri, Muhammad Asif : Automatic Number Plate Recognition System for Vehicle Identification Using Optical Character Recognition, International Conference on Education Technology and Computer, IEEE 2009.

[11] Osslan Osiris Vergara Villegas, Daniel Gonzál ez Balderrama, Humberto de Jesús Ochoa Domínguez and Vianey Guadalupe Cruz Sánchez,” License Plate Recognition Using a Novel Fuzzy Multilayer Neural Network”, International Journal of Computers, Issue 1, Volume 3, 2009, pp.31-40.

[12] P.Anishiya, Prof. S. Mary Joans, “Number Plate Recognition for Indian Cars using Morphological Dilation and Erosion with the Aid of Ocrs” , 2011 International Conference on Information and Network Technology IACSIT Press, Singapore, IPCSIT vol.4 2011, pp.116-119.

[13] Prof. Thomas B. Fomby – “K-Nearest Neighbors Algorithm: Prediction and Classification” Department of Economics Southern Methodist University Dallas, TX 75275 February 2008

[14] R. C. Gonzalez and R. E. Woods, Digital Image Processing, third ed., New Jersey: Prentice Hall, 2008.

[15] Rainer Lienhart and Axel Wernicke. Localizing and segmenting text in images and videos. IEEE Transaction On Circuits and Systems for Video Technology, 12(4), April 2002.

[16] Raveendran Pillai, Prot: (Dr). Sukesh Kumar. A: A Real-time system for the automatic identification of motorcycle - using Artificial Neural Networks Recognition, World Academy of Science, Engineering and Technology 9, 2005.

[17] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, “Automatic license plate recognition (ALPR): A state-of-the-art review,” IEEE Trans. Circuits Syst. Video Technol., vol. 23, no. 2, pp. 311-325, 2013.

[18] Christos Nikolaos E. Anagnostopoulos, Ioannis E. Anagnostopoulos, Vassili Loumos, and Eleftherios Kayafas, “A License Plate-Recognition Algorithm for Intelligent Transportation System Applications”, IEEE, Transactions on Intelligent Transportation Systems. Vol. 7, No.3, September 2006. pp. 377-392.

[19] Danian Zheng, Yannan Zhao, and Jiaxin Wang. An efficient method of license plate location. Pattern Recogn.Lett., 26(15):2431–2438, 2005.

[20] F. Aghdasi and H. Ndungo, “Automatic licence plate recognition system,”

IEEE AFRICON Conf., vol. 1, pp. 45–50, 2004.

[21] Hamid Mahini, Shohreh Kasaei, Faezeh Dorri, and Fatemeh Dorri. An efficient features based license plate localization method. In ICPR’06: Proceedings of the 18thInternational Conference on Pattern Recognition, pages 841–844, Washington, DC, USA, 2006. IEEE Computer Society.

[22] J. Sauvola and M. Pietikäinen, “Adaptive document image binarization,” Pattern Recognit., vol. 33, p. 225-236, 2000.

[23] J.-M. Guo and Y.-F. Liu, “License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques,” IEEE Trans. Veh. Technol., vol. 57, no. 3, pp. 1417-1424, 2008.

[24] L. Angeline, W. Y. Kow, W. L. Khong, M. Y. Choong, and K. T. K. Teo, “License plate character recognition via signature analysis and features extraction,” Proc. Int. Conf. Comput. Intell., Model. Simulation, pp. 1-6, 2012.

[25] Audibert, J.Y. & Tsybakov, A.B. (2007) "Fast learning rates for plugin classifiers under the margin condition", Ann. Statist, 35: 608–633.

[26] Bailey, T. & Jain, A. (1978) "A note on distance-weighted k-Nearest Neighbor rules", IEEE Trans. Systems, Man, Cybernetics, 8: 311-313.

[27] Baoli, L., Shiwen, Y. & Qin, L. (2003) "An Improved k-Nearest Neighbor Algorithm for Text Categorization, ArXiv Computer Science e-prints.

[28] Bauer, M.E., Burk, T.E., Ek, A.R., Coppin, P.R. Lime, S.D., Walsh, T.A., Walters, D.K., Befort, W. & Heinzen, D.F. (1994) "Satellite Inventory of Minnesota’s Forest Resources", Photogrammetric Engineering and Remote Sensing, 60(3): 287–298.

[29] Bax, E. (2000) "Validation of nearest neighbor classifiers", IEEE Trans. Inform. Theory, 46: 2746–2752.

[30] Benetis, R., Jensen, C., Karciauskas, G. & Saltenis, S. (2006) "Nearest and Reverse Nearest Neighbor Queries for Moving Objects", The International Journal on Very Large Data Bases, 15(3): 229–250.

[31] Bermejo, T. & Cabestany, J. (2000) "Adaptive soft k-Nearest Neighbor classifiers", Pattern Recognition, 33: 1999-2005.

[32] Chitra, A. & Uma, S. (2010) "An Ensemble Model of Multiple Classifiers for Time Series Prediction", International Journal of Computer Theory and Engineering, 2(3): 1793-8201.

[33] Cover, T.M. (1968) "Rates of convergence for nearest neighbor procedures", In Proceedings of the Hawaii International Conference on System Sciences, Univ. Hawaii Press, Honolulu, 413–415.

[34] Cover, T.M. & Hart, P.E. (1967) "Nearest neighbor pattern classification", IEEE Trans. Inf. Theory, 13: 21–27.

[35] Devroye, L. (1981) "On the asymptotic probability of error in nonparametric discrimination", Ann. Statist, 9: 1320– 1327.

[36] Devroye, L., Gyorfi, L., Krzyzak, A. & Lugosi, G. (1994) "On the strong universal consistency of nearest neighbor regression function estimates", Ann. Statist, 22: 1371– 1385.

[37] Devroye, L. & Wagner, T.J. (1977) "The strong uniform consistency of nearest neighbor density estimates", Ann. Statist., 5: 536–540.

[38] Devroye, L. & Wagner, T.J. (1982) "Nearest neighbor methods in discrimination, In Classification, Pattern Recognition and Reduction of Dimensionality", Handbook of Statistics, 2: 193–197. North-Holland, Amsterdam.

[39] Domeniconi, C., Peng, J. & Gunopulos, D. (2002) "Locally adaptive metric nearest neighbor classification", IEEE Transactions on Pattern Analysis and Machine Intelligence. 24(9): 1281–1285.

[40] Dudani, S.A. (1976) "The distance-weighted k-nearest neighbor rule", IEEE Transactions on System, Man, and Cybernetics, 6: 325-327.

[41] Eldestein, H.A. (1999) "Introduction to Data Mining and Knowledge Discovery", Two Crows Corporation, USA, ISBN: 1-892095- 02-5.