****

**T.E. Project Report**

**On**

**AUTOMATIC VEHICLE NUMBER PLATE IDENTIFICATION & ITS TRANSMISSION TO CLOUD**

**Submitted by,**

**Pranav Gundewar (T120023045)**

**Hrishikesh Hippalgaonkar (T120023049)**

**Vihang Khare (T120023156)**

**Project Guide:**

**Dr. Vinaya Gohokar (Internal Guide)**

**TE Div: I**

**Year: 2015-2016**

**Maharashtra Institute of Technology, Pune – 38.**

**Department of Electronics and Telecommunication**

**CERTIFICATE**

MAEER’s

MAHARASHTRA INSTITUTE OF TECHNOLOGY, PUNE.



This is to certify that the project entitled

**AUTOMATIC VEHICLE NUMBER PLATE IDENTIFICATION & ITS TRANSMISSION TO CLOUD**

has been carried out successfully by

**Pranav Gundewar (T120023045)**

**Hrishikesh Hippalgaonkar (T120023049)**

**Vihang Khare (T120023156)**

during the Academic Year **2015-2016** in partial fulfillment of their

course of study for third year in ***Electronics and***

***Telecommunication*** as per the syllabus prescribed by the

**University of Pune.**

**Dr. Vinaya Gohokar Prof. Dr. G. N. Mulay**

**Internal Guide Head of Department**

(Electronics & Telecommunication)

MIT, Pune

**ACKNOWLEDGEMENT**

A project is an opportunity for the students to practically implement theoretical concepts. It proves to be a learning platform for the students so that they can compete successfully in their professional life. However, in this entire journey of completing the project, we need proper guidance so as to avoid obvious mistakes.

We would like to thank our principal Prof. Dr. L. K. Kshirsagar for his constant encouragement. We would also like to thank our Head of E&TC Dept., Prof. Dr. G. N. Mulay.

We would thank our internal guide Prof. Dr. V. Gohokar for her invaluable guidance & support for making this project a success.

Lastly, our sincere thanks to all the staff members & our friends who helped us in all possible ways to make this project a success.

**ABSTRACT**

Number plate recognition was invented in 1976 at the Police Scientific Development Branch in the UK. It is a mass surveillance method that uses optical character recognition(OCR) on images to read vehicle registration plates. Automatic Number plate recognition is a real time embedded system which identifies the characters directly from the image of the vehicle number plate. Due to rapid development of technology and increasing use of vehicles, number plate recognition has become necessary and it is playing an important role in variety of applications related to automated transport system such as road traffic monitoring, detection of stolen vehicles, automatic payments of tolls on highways or bridges, parking lots access control etc. Number plate recognition system is composed of three parts, which are Number plate detection, character segmentation and recognition. Systems for Automated Number Plate Recognition (ANPR) can provide a valuable data source for transport planning and engineering. Multiple tasks can be solved with ANPR systems located at one or more survey points. The resulting data is then used to compare with the records on a database so as to come up with the specific information like the vehicles owner, place of registration, address etc.

**TABLE OF CONTENTS**

1. **Introduction**………………………………………………………………………..1
   1. Introduction…………………………………………………………………….1
   2. Aim………………………………………………………………………….….2
   3. Basic Scope and Background……………………………………………….….2
   4. Objectives………………………………………………………………………2
2. **Survey**………………………………………………………………………………3
   1. Literature Survey……………………………………………………………….3
3. **System Design**……………………………………………………………………...4
   1. System Specifications……………………………………………………….….4
   2. System Block……………………………………………………………...……4
   3. Description………………………………………………………………...…....5
      1. Camera……………………………………………………………….….5
      2. Raspberry Pi………………………………………………………….….5
   4. Experimental Setup………………………………………………………...…...5

3.4.1. Motion Detection using IR Sensor………………………………………6

3.5. Component and Devices…………………………………………………….......6

3.6. Selection of Components…………………………………………………….….7

3.6.1. Choice of Microprocessor…………………………………………….…7

3.6.2. Choice of Camera……………………………………....………………..9

3.7. Circuit Diagram………………………………………………………………...10

4. **System Development**………………………………………………………………11

4.1. Hardware Model……………………………………………………………….11

4.2. Software Model………………………………………………………………...11

4.3. System Software………………………………………………………………..16

4.3.1. Flowchart……………………………………………………………....17

4.3.2. Algorithm………………………………………………………………18

5. **Testing and Debugging**……………………………………………………………19

5.1. Simulation………………………………………………………………………19

5.2. PCB Layout……………………………………………………………………20

6. **Result**……………………………………………………………………………...22

7. **Conclusion**…………………………………………………………………………23

8. **Bill of Materials**…………………………………………………………………...24

9. **Future scope and applications**……………………………………………………25

10. **Bibliography**……………………………………………………………………….26

**Appendix**

Datasheet of IR LED Transmitter TSFF5210 & receiver BPV10NF

Datasheet of LM 358

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Name of Figure** | **Page No.** |
| 3.2 | Block diagram | 4 |
| 3.4.1 | Experimental setup block diagram | 5 |
| 3.4.2 | Experimental setup | 6 |
| 3.5 | IR Transmitter and Receiver circuit | 10 |
| 4.2.1 | Grey Image | 12 |
| 4.2.2 | Output of Gaussian filter | 12 |
| 4.2.3 | Output of Sobel filter | 12 |
| 4.2.4 | Output of Threshold filter | 13 |
| 4.2.5 | Image with validated and invalidated contours | 14 |
| 4.2.6 | Mask images created by various seeds passed to *floodfill()* function | 15 |
| 4.3.1 | OpenCV logo | 16 |
| 4.3.2 | Python logo | 16 |
| 4.3.1.1 | Flowchart | 17 |
| 5.1.1 | Simulation circuit (Obstacle present) | 19 |
| 5.1.2 | Simulation circuit (Obstacle not present) | 19 |
| 5.2.1 | Schematic diagram of circuit on Eagle Software | 20 |
| 5.2.2 | Board layout of circuit on Eagle Software | 20 |
| 5.2.3 | Track side PCB layout of circuit | 21 |
| 5.2.4 | Component side PCB layout of circuit | 21 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of table** | **Page No.** |
| 1. | Comparison of Microprocessors and on-board features | 7 |
| 2. | Comparison of Peripheral utilities | 8 |
| 3. | General purpose I/O | 9 |
| 4. | Choice of camera | 9 |

**CHAPTER 1 INTRODUCTION**

**1.1 INTRODUCTION**

Automobile number plate recognition and extraction is an image-processing technology used to identify automobiles by their license plates. The vehicle number plate recognition was invented in 1976 at the Police Scientific Development Branch in the UK. In a sense, the automobile number plate recognition with OCR is a combination of integrated hardware and software that will read vehicle license plates without the need of humans to do it.

The concept of the system is to identify properly and locate the vehicle they are looking for. It is a type of technology, mainly software that enables computer systems to read automatically the registration number (license number) of vehicles from digital pictures. Systems use infrared lighting to help the camera take the picture at any time during the day. This combination works fine during day and night and provides constant good image quality. The system is computationally inexpensive compared to the other ANPR systems Besides the robustness, the earlier methods use either feature based approached using edge detection or Hough transform which are computationally expensive or use artificial neural network which requires large training data.

Internet connection is provided to the Raspberry Pi for sending files to cloud. The ANPR system works in three steps, the first step is the detection and capturing a vehicle image, the second step is the detection and extraction of number plate in an image using Digital Image Processing. Digital Image processing refers to the processing of digital images by means of digital computer. The third section uses image segmentation technique to get individual character and optical character recognition (OCR) to recognize the individual character with the help of Tesseract (open source OCR engine). Output is then transmitted to a cloud (Google Drive).

Interest in Digital Image Processing stems from 2 principal application areas:

1. Improvement of pictorial information for data interpretation.
2. Processing of data image for storage, transmission & representation.

**1.2 AIM**

Development of ANPR system to identify vehicles by their number plates using Optical Character Recognition (OCR) & transmitting output file to the cloud (Google Drive).

**1.3 BASIC SCOPE AND BACKGROUND**

This project first step to make automated ANPR system in low cost.

The aim is to demonstrate vehicle motion detection and obtaining vehicle registration plate & comparing with standard database available worldwide. A small database for particular vehicles can be made for a particular plot like parking space etc. The obtained data can be transmitted to cloud to have remote monitoring access.

In future, ANPR system can be made foolproof by having access to a person’s database like contact no, address etc. Further, incorporating ANPR system with currently installed cameras (CCTV) will improve functionality & usability of available system.

**1.4 OBJECTIVES**

Main objective is to develop hardware system such as camera interface with microprocessor. In order to develop proposed system, study & implementation of algorithm for detection of number plate using OCR is studied and best feasible algorithm is to be selected.

The final & important part of proposed system is to transmit processed data like output file to a cloud (Google Drive).

**CHAPTER 2 SURVEY**

**2.1 LITERATURE SURVEY**

While researching about how to detect characters, digits & symbols from different images (frames), we came across several research papers regarding real-time embedded system for detection of number plate & its character recognition.

Among the various algorithms, optical character recognition (OCR) is best suited & useful algorithm is selected which is basically an algorithm where frame by frame analysis by applying various image processing functions is carried out & characters are recognized successfully.

The summary of research papers we referred is given as follows-

Raja Vikramdeep Singh, Navneet Randhawa explained use of OCR in recognizing characters from vehicle number plate [1]. The concept of 4 main steps of recognition and their respective output valuation is well explained. The developed algorithm has successfully reduced complexity of algorithm in sense of speed & time constraint.

M. T. Qadri, Muhammad Asif presented two sections of system comprising hardware & software models of developed ANPR system [2]. The row & column segmentation method has been explained. Results shows that the system robustly detect and recognize the vehicle using license plate against different lightening conditions. However, the camera used in this project is sensitive to vibration and fast changing targets due to the long shutter time.

Er. Kavneet Kaur, Vijay Kumar Banga have proposed a two stage hybrid recognition system combining statistical and structural recognition method [3]. The Template matching affects the accuracy of number plate recognition. Some factors which affect the effectiveness of template matching based on OCR technique i.e. font type, noise in image, tilting etc. have been found out. However, work can be done on these factors and efficiency may be increased.  
**CHAPTER 3 SYSTEM DESIGN**

**3.1 SYSTEM SPECIFICATIONS**

The idea of the project is to develop a system to detect characters on vehicle number plate. Raspberry Pi has been used to develop the software aspect of the system. The algorithm has to be robust and feasible to be implemented in real time applications. Internet connection is provided to Pi for connecting it with cloud.

***Hardware:***

* Raspberry Pi 2 Board (Model B)
* Power Supply: 3.3/5V
* Logitech camera resolution: 3 MP
* IR LED Transmitter TSFF5210 & receiver BPV10NF

***Software:***

* Raspberry Pi (with Raspbian OS)
* OpenCV
* Tesseract OCR Engine [4]
* Eagle v7.5
* Multisim

***Platform / Language used:***

* Python

**3.2 SYSTEM BLOCK DIAGRAM**

The general system block diagram can be shown to be as-

Monitor

Raspberry Pi

Camera

Cloud

Keyboard

Fig 3.2: Block Diagram

**3.3 DESCRIPTION**

**3.3.1 CAMERA**

The camera is interfaced with the microcontroller i.e. Raspberry Pi. Input image is taken from cameras like CCTV, webcam. The captured image is then sent to Pi. The camera is arranged at a distance about 1-2 meters to get good focused image of car which will ensure number plate is included in the image.

**3.3.2 RASPBERRY PI**

Raspberry Pi is a microprocessor which is used for image & data processing. It acts as platform between user & sensors, modules etc. Our attempt is to minimize cost of OCR system as well as reducing human intervention.

The image captured by camera is transferred to Pi which processes the image & sends the output file to cloud i.e. virtual storage (Google Drive). The ouput image is then stored on drive to have remote access available for user.

Keyboard & monitor are used for programming the Pi.

**3.4 EXPERIMENTAL SETUP**

IR TX

Raspberry Pi

Camera

Vehicle

IR RX

Fig 3.4.1 Experimental Setup Block Diagram

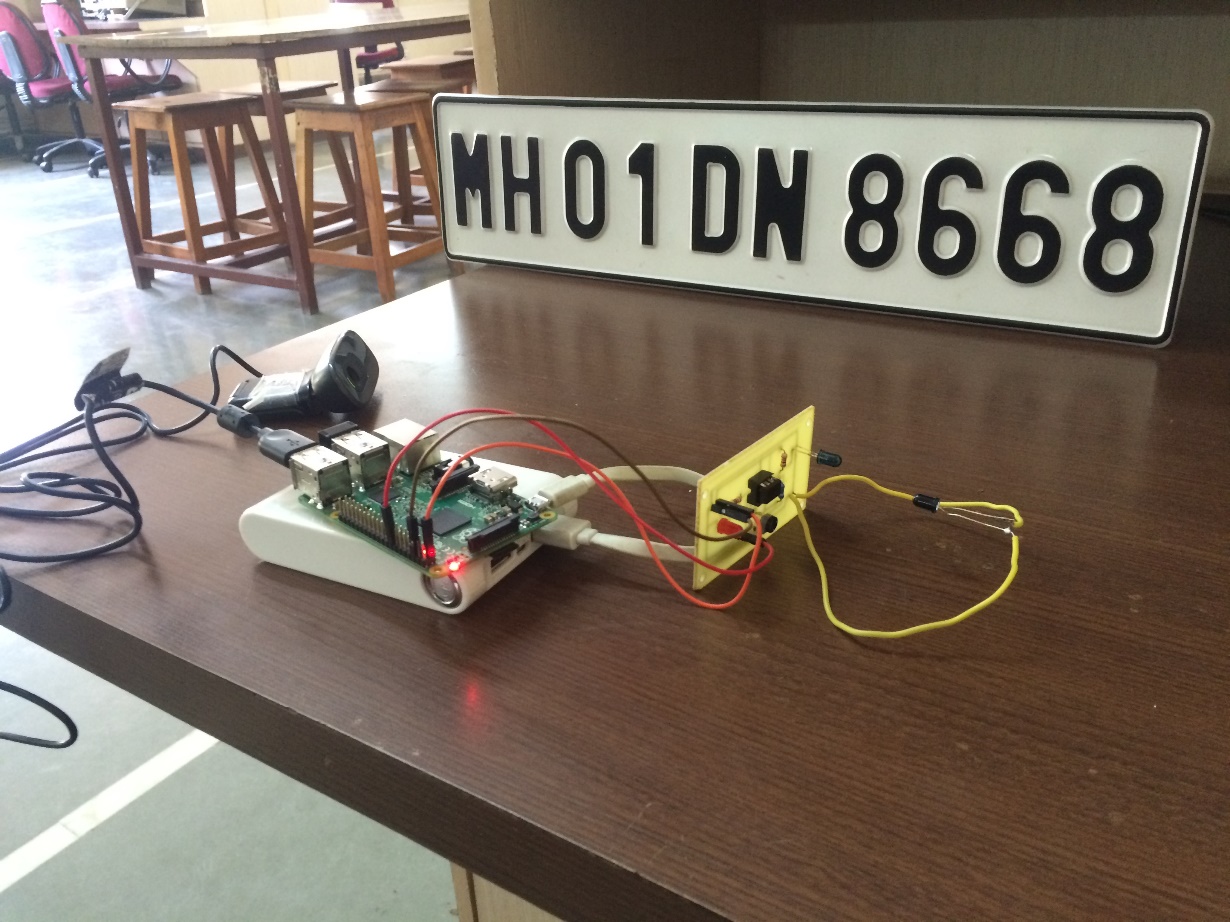


Fig 3.4.2 Experimental Setup

**3.4.1 MOTION DETECTION USING IR SENSOR**

The car approaches the IR sensor in the driveway. As the car passes through the IR sensor, a signal is sent to Raspberry Pi to activate the camera after a stipulated amount time.

After the stipulated time, the camera captures an image of the number plate of the car.

The captured image is then processed on the Pi using various OpenCV functions.

**3.5 LIST OF COMPONENTS AND DEVICES**

1. IR LED Transmitter TSFF5210 & receiver BPV10NF
2. Resistors

* 220E
* 1K
* 6.8K
* 10K potentiometer

1. Logitech C270 camera:

* Photos: Up to 3.0 megapixels (software enhanced)
* Logitech Fluid Crystal™ Technology
* Hi-Speed USB 2.0 certified
* Video capture: Up to 1280 x 720 pixels

1. Raspberry Pi 2 (Model B)

**3.6 SELECTION OF COMPONENTS**

**3.6.1 CHOICE OF MICROPROCESSOR**

Raspberry Pi is popular & advanced microprocessor used all over the world. Moreover, it can be used to solve numerous daily problems & can be easily used in lots of projects. Main advantages of Pi includes its compatibility with various USB devices & Arduino Uno development board.

Table 1: Comparison of Processors & on-Board Features

|  |  |  |
| --- | --- | --- |
|  | **Galileo 2** | **Raspberry Pi 2 Model B** |
| **Processor** | Intel® Quark X1000 – single core | **BCM2836 (ARMv7) - Quad Core** |
| **Description of Processor** | Quark, described by Intel at IDF2013, is very low power consumption, small form factor, and low cost; ideal for “wearables,” and the Internet of Things†. | BCM2836 is a power efficient, full HD, multimedia applications processor for media playback, imaging, camcorder, streaming media, graphics and 3D gaming. |
| **Architecture** | Intel ® Pentium® Class | **ARM® Cortex-A7** ARM1176JZ-F |
| **Speed** | 400MHz | **900MHz** |
| **Real Time Clock** | Yes, needs a 3.3v coin cell | No |
| **Cache** | 16 KB L1 cache | **Dedicated 512 KB CPU cache** |
| **RAM** | 512KB on-chip SRAM & 256MB DRAM | **1GB LPDDR2 SDRAM** |
| **FLASH memory** | 8MB NOR Flash for FW bootloader & sketch storage | No permanent on-board Flash memory. |
| **EEPROM** | 11KB | No, but Pi 2 easily accommodates an external EEPROM. |

**Table 2: Comparison of Peripherals/Utilities**

|  |  |  |
| --- | --- | --- |
|  | **Galileo 2** | **Raspberry Pi 2 Model B** |
| **Analog I/O** | Up to 6 Analog Inputs with 10- or 12-bit (programmable) resolution. A0 – A5 can be used as Digital I/O using functions and operate at 3.3V or 5V. Each pin sources or sinks 16mA @ 5v or 8mA @3.3v | 26 general purpose I/O (GPIO) pins (access to I2C, UART, and SPI) on a 40-pin header. Analog input using an ADC must be an additional purchase of an external companion board |
| **Digital I/O** | 14 Digital I/O that can be used as input or output and operate at 3.3V or 5V. Each pin sources or sinks 16mA @ 5v or 8mA @3.3v | Identical to RPi B+. Up to 17 GPIO pins that can be programmed as Digital Input or Output. |
| **PWM** | Up to 6 of the DI/O can be configured as 8- or 12-bit Pulse Width modulation (PWM). Programmable resolution. | Two pins at a time may be set up as PWM. One is shared with audio. |

**Table 3: General Purpose I/O**

|  |  |  |
| --- | --- | --- |
|  | **Galileo 2** | **Raspberry Pi 2 Model B** |
| USB 2.0 | 2 ports: Native USB 2.0 Host (standard Type A) & USB 2.0 Client (micro-USB Type B.) 3rd USB Host available over mPCIe. | 4 USB ports. Managed by the LAN9514 USB/Ethernet controller. |
| Ethernet (RJ45) | 10/100 Mbps with a dedicated PHY for Ethernet control. One RJ45 port. Does not require use of SPI. Power over Ethernet capable. | 10/100 Mbps Ethernet via RJ45 port. Managed by the LAN9514 USB/Ethernet controller |

Top of Form

**3.6.2 CHOICE OF CAMERA**

|  |  |  |
| --- | --- | --- |
| **PARAMETERS** | **LOGITECH C270** | **IBALL FACE2FACE C12.0** |
| Dimension | 21 x 7.6 x 15.2 cm | 21 x 8 x 16 cm |
| Sensor | CMOS | ¼ CMOS |
| Image Resolution | 3 MP | 2 MP |
| Video Quality | 720p | 480p |
| Frames Per Second | 50 | 15 |
| Interface | USB 2.0 | USB 2.0 |
| Cost | INR 1000/- | INR 800/- |

**3.7 CIRCUIT DIAGRAM**

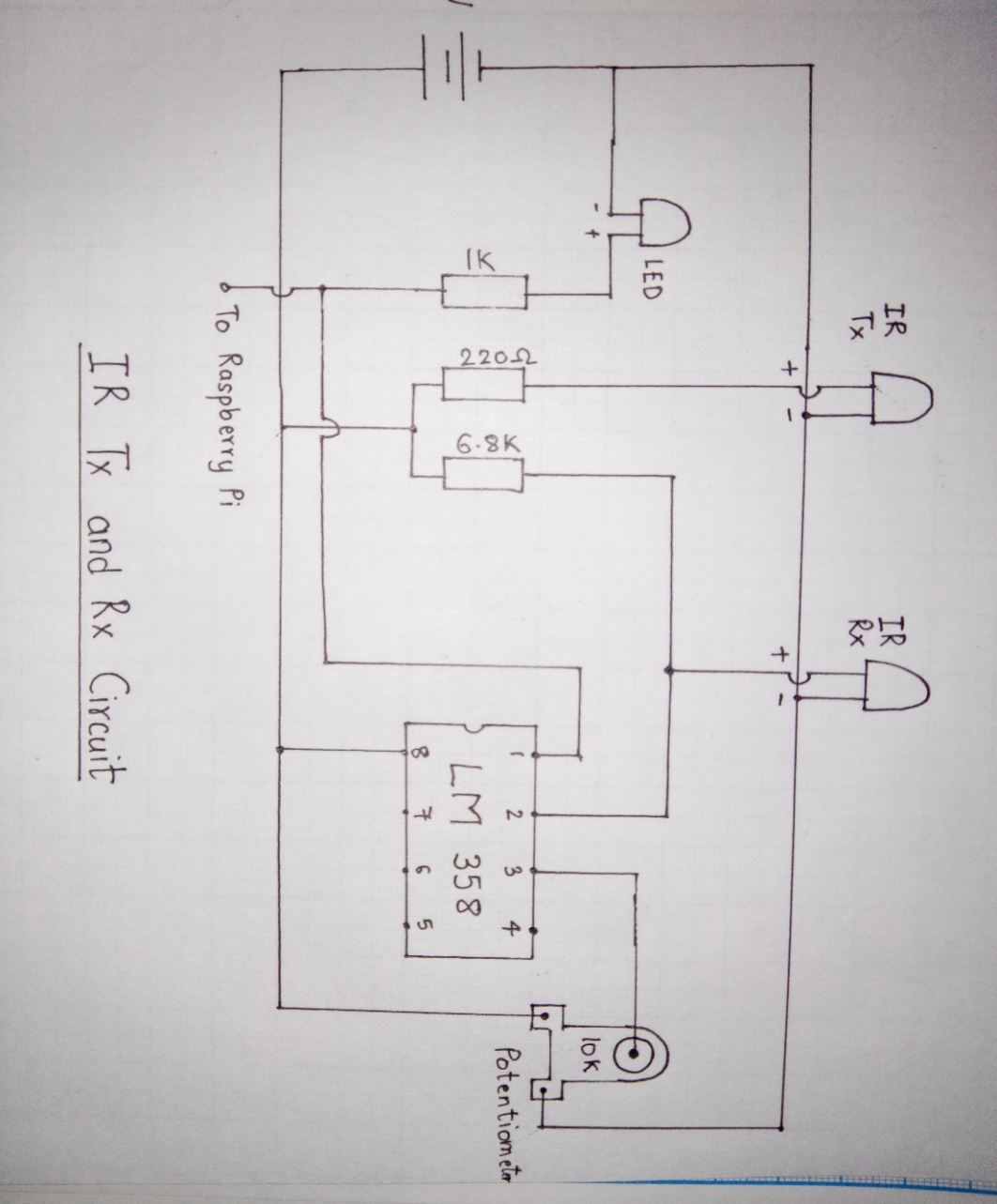
****

Figure 3.5: IR Transmission and Reception Circuit

**CHAPTER 4 SYSTEM DEVELOPMENT**

**4.1 HARDWARE MODEL**

The hardware portion of the project includes the IR circuit.The IR sensor is connected to one of the GPIO pins of the Raspberry Pi.The Raspberry Pi monitors the signal from the sensor for a falling edge in the signal,which indicates a car has passed by the sensor and is coming close to the area in the field of vision of the camera.

**4.2 SOFTWARE MODEL**

The main and the most important portion of this system is the software model. The software model use series of image processing techniques which are implemented in Python & OpenCV. The ANPR algorithm is broadly divided into three parts:

• Capture image

• Extract the plate from the image

• Recognize the numbers from the extracted plate

1. **Capture image**

* Once the Raspberry Pi detects the falling edge of signal from the IR circuit, the camera captures an image of the car after a specified interval of time.
* The time interval depends on the rough estimation of time required for the car to reach from the area of camera after passing by the IR sensor.
* The captured image is temporarily stored in the memory card on Raspberry Pi.

1. **Extract the plate from the image**

* The captured image is then subjected to Segmentation.Segmentation is the process of dividing an image into multiple segments. This process is to simplify the image for analysis and make feature extraction easier.
* The captured image is converted to grayscale using the function (*cv2.cvtColor),* as a colored image won’t be of any use. Then the gray-scaled image is passed through a 5x5 Gaussian blur filter, using the function (*cv2.GaussianBlur)*, to remove noise in the image. (Fig. 4.2.1 & Fig 4.2.2)



Fig. 4.2.1: Grey Image Fig 4.2.2: Output of Gaussian Blur

* One important feature that can be exploited from number plates are the high number of vertical edges. To find the vertical edges, a Sobel filter is used (*cv2.Sobel*) and its first horizontal derivative is found out. (Fig. 4.2.3)



Fig. 4.2.3: Output of Sobel Filter

* After a Sobel Filter, a threshold filter is applied to obtain a binary image with a threshold value obtained through Otsu's Method. Otsu's algorithm needs an 8-bit input image and Otsu's method automatically determines the optimal threshold value (*cv2.threshold*). (Fig. 4.2.4)

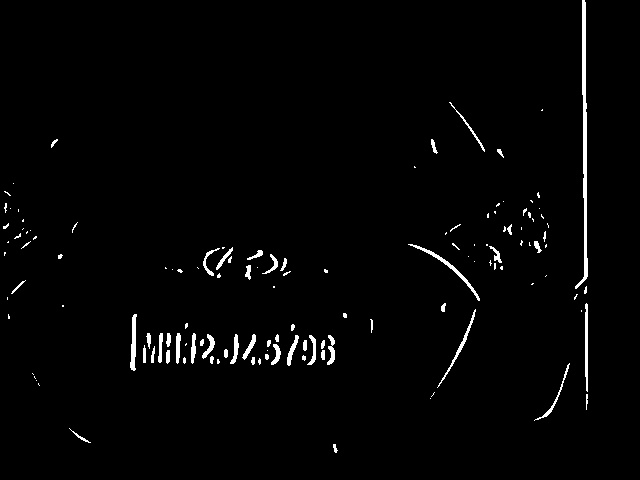


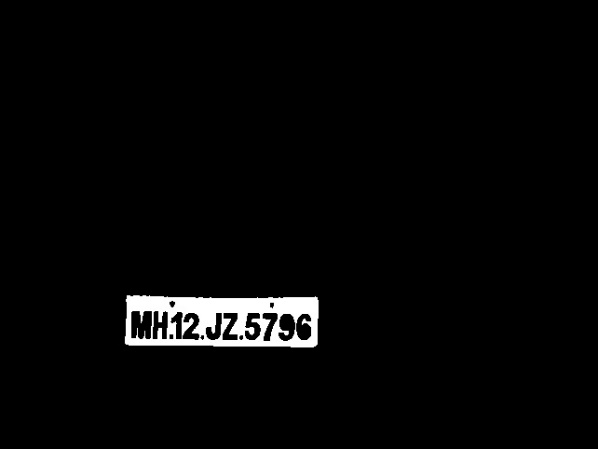
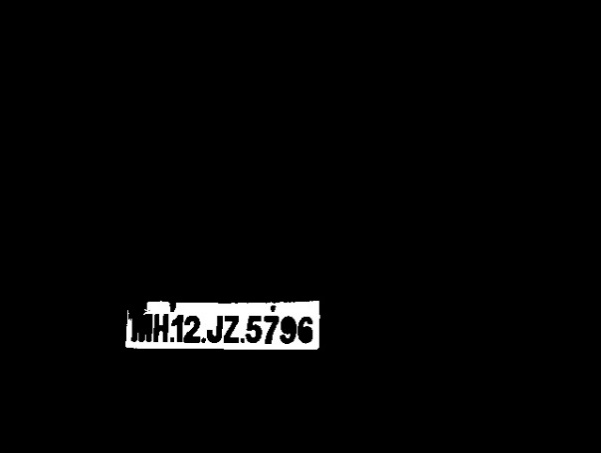
Fig 4.2.4: Output of Threshold Filter

* By applying a close morphological operation (*cv2.morphology(, cv2.MORPH\_CLOSE, )*), blank spaces between each vertical edge line are removed, and all regions that have a high number of edges are connected. In this step the possible regions that can contain plates are obtained.
* After all these pre-processing steps, the regions in the image that have the possibility of containing license plates are obtained.This calls for the use of (*cv2.findContours***)** function. This function retrieves the contours of a binary image, and only the external contours are needed.
* Now all the detected contours approximated to their rectangular Bounding-Boxes can be visualized. OpenCV has *(cv2.****minAreaRect)*** function exactly for this task. The (cv2.***BoxPoints)*** function can be used to extract all the four co-ordinates of the rectangle which is then used to draw the Bounding-Box.
* To remove the false positive contours, the outliers can be validated against an assumed area and aspect ratio. Once the validation is complete only a few contours remain, one of which includes the number plate. (Fig 4.2.5)



Fig. 4.2.5: Image with validated(Red) and invalidated(Green) contours

* One more feature that can be exploited from number plate is that they have a white background, so ‘*floodfill’* algorithm can be used.***‘floodfill’*** is very similar to the old *‘Fill Color’* that one comes across in MS Paint or other drawing programs. It tries to spread the chosen color from the point of origin to every direction until it faces a tangible boundary preventing it to go any further.
* For applying this algorithm, the origin (also known as ***seeds***) must be known. Since how to choose a specific ***seed*** within these validated contours is not known, the *seeds* will be randomized, hoping that at least one of them succeeds in exploiting the white background of an actual License Plate and *‘****floodfills’*** a considerable chunk of it.
* These randomly generated *seeds* are then passed to the (*cv2.floodfill*) function which creates a new mask image for each *seed.* (Fig.4.2.6)
* Finally, at least one of the mask images will contain just the number plate.

****

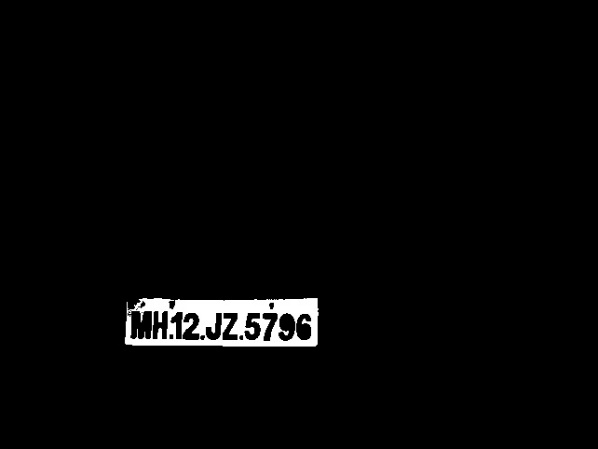
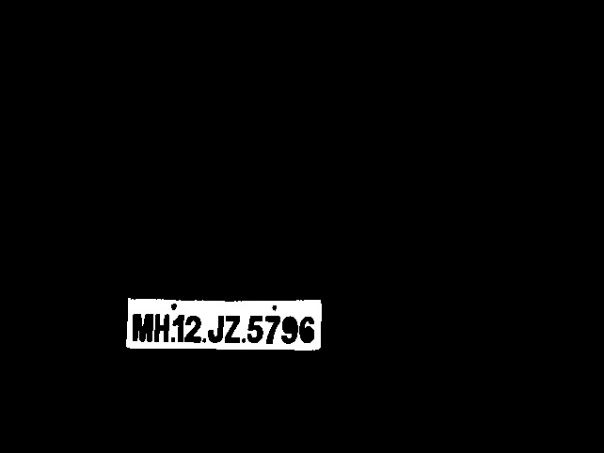
****

Fig. 4.2.6: Mask images created by various seeds passed to *floodfill()* function

1. **Recognize the numbers from the extracted plate:**

* All the mask images, original captured image are then saved on to Google Drive cloud, which is synced with the Raspberry Pi. Thus, these images can then be accessed and viewed from anywhere
* Now all the mask images are passed to *Tesseract OCR engine* irrespective whether all of them contain number plate or not.
* *Tesseract* is an optical character recognition engine for various operating systems. It is free software, released under the Apache License, Version 2.0, and development has been sponsored by Google since 2006.*Tesserac*t is considered as one of the most accurate open source OCR engines currently available.
* *Tesseract* analyzes the mask images and provides the number plate text output only for those images which have the number plate in them.
* The number plate text is saved automatically as a text document (.txt) in the Google Drive Cloud.

**4.3 SYSTEM SOFTWARE**

For implementation of the algorithm of preprocessing & OCR, we used the software Python & OpenCV. The software is able to perform frame-by-frame analysis of the image. The software is also able to process the images rapidly in real time thus enabling accurate motion detection.

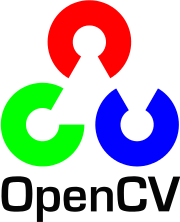


Fig. 4.3.1 OpenCV Logo Fig. 4.3.2 Python Logo

**4.3.1 FLOWCHART**

Do Image Processing

**START**

Raspberry Pi takes input from obstacle detection circuit

Check for high to low pulse

Capture image of license plate

Check for valid contours in image and select the best image

Give image as input to Tesseract

Output of tesseract is a text file containing the number plate text

Upload it to Google Drive

Yes

Yes

**END**

Fig 4.3.1.1 Flowchart

**4.3.2 ALGORITHM**

* IR sensor is used for motion detection. Here, we have used IR sensors for detection of obstacle i.e. vehicle.
* When a vehicle approaches driveway, IR sensor detects its movement & changes its output. This output is given to Raspberry Pi.
* Pi detects this signal & camera interfaced with it captures the image of the vehicle.
* Captured image is then given to Pi & Pi processses the captured image.
* Pi performs quite a few Image Processing operations mainly –
* The image is then converted to a grey scale version.
* The image is blurred in order to lessen the effect of noise.
* Countours are detected & validated.
* Floodfill algorithm is used to mask the valid countour containing the number plate .
* This processed image is then given to Tesseract which is an open source OCR engine.
* Tesseract recongnizes the chararters & symbols from image. It creates a text file in which output is stored.
* Output file is then uploaded to Google Drive for database applications.

**CHAPTER 5 TESTING AND DEBUGGING**

**5.1 SIMULATION**

For simulation part, we have used Multisim software which is free software.

We have used Multisim because its ease of use & vast library components available. Simulation includes 2 conditions- when obstacle is detected & when there is no obstacle.

1. When obstacle is detected:

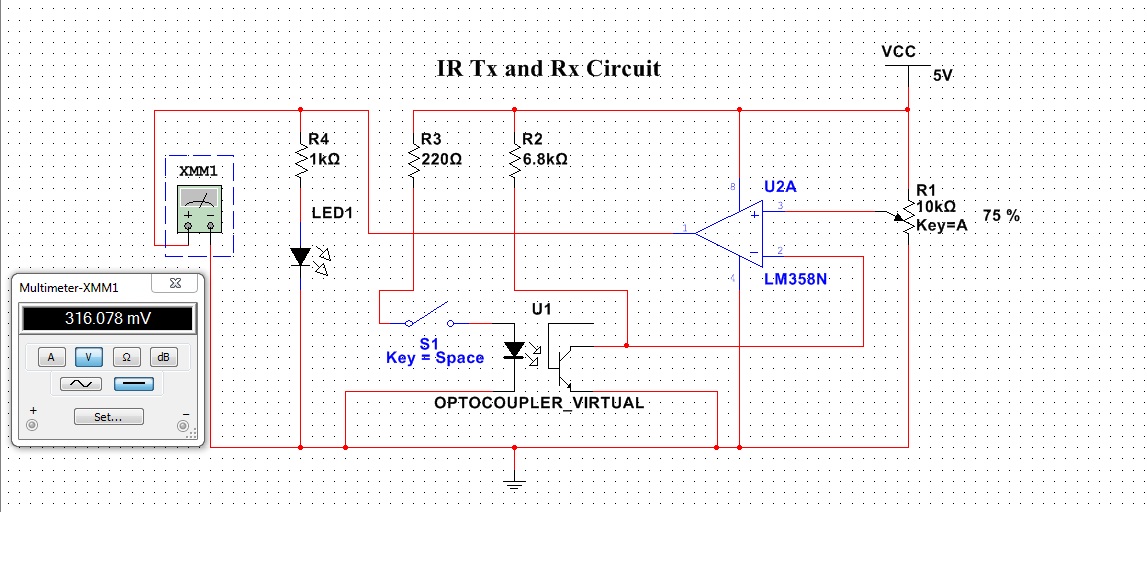


Fig 5.1.1: When obstacle is detected

1. When there is no obstacle:

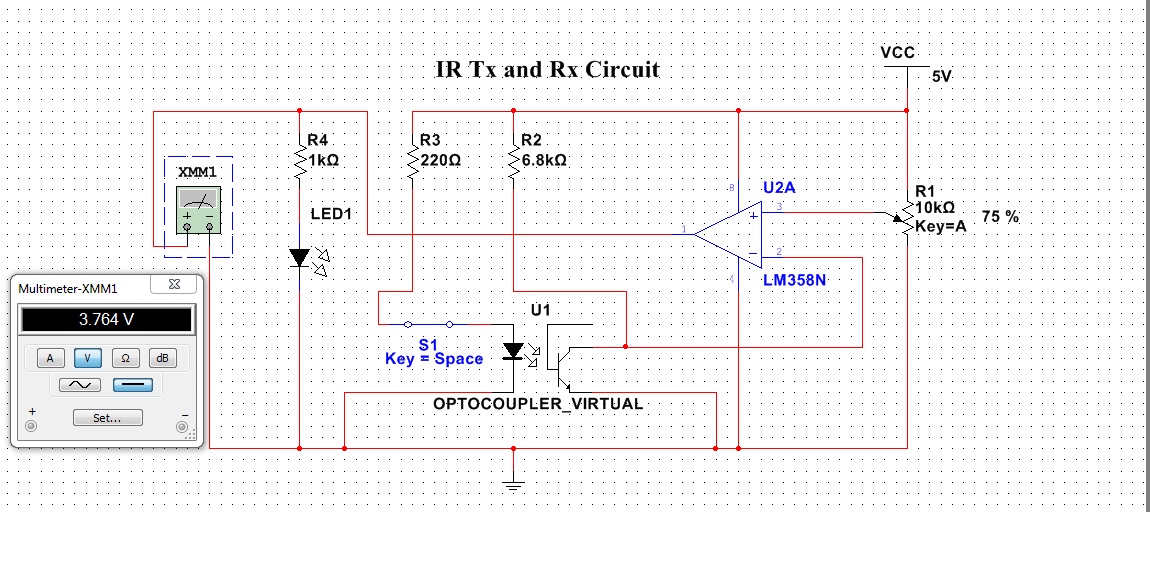


Fig 5.1.2: When there is no obstacle

**5.2 PCB LAYOUT**

PCB layout is done using Eagle v7.5. This sofware is very easy to understand & have large components library.

1. Schematic:

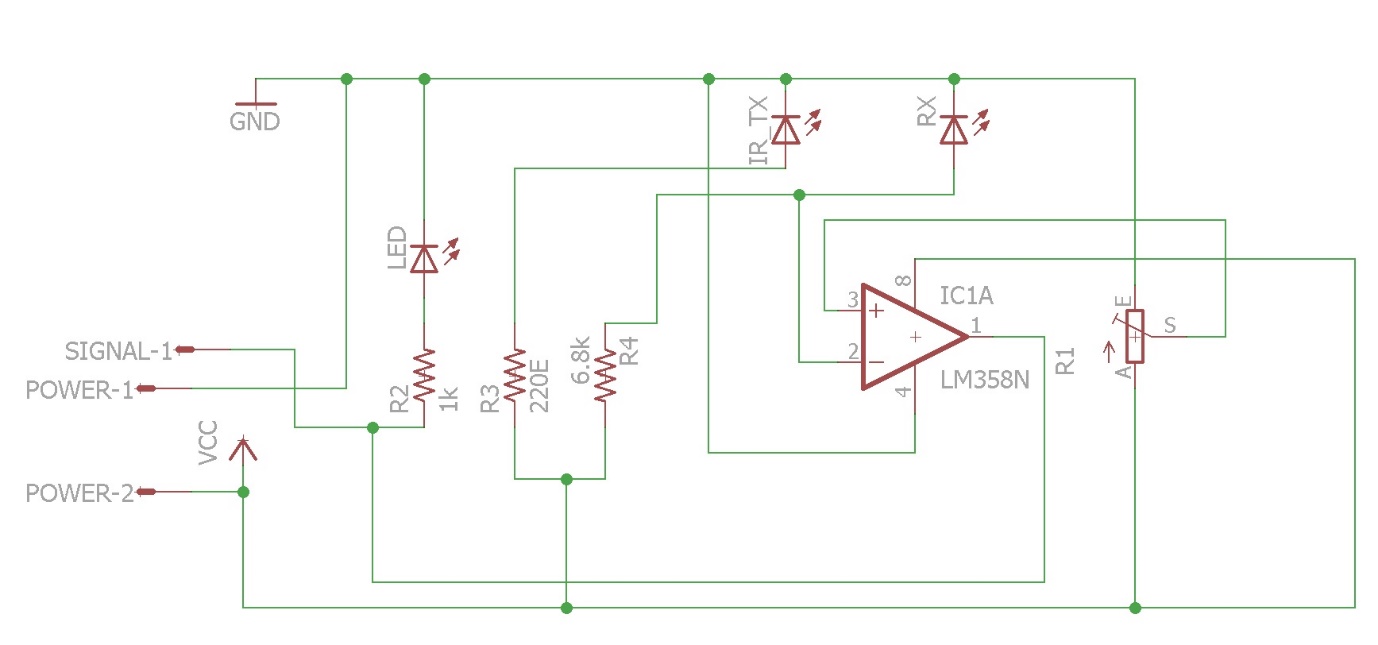


Fig 5.2.1 Schematic diagram of circuit

1. Board:

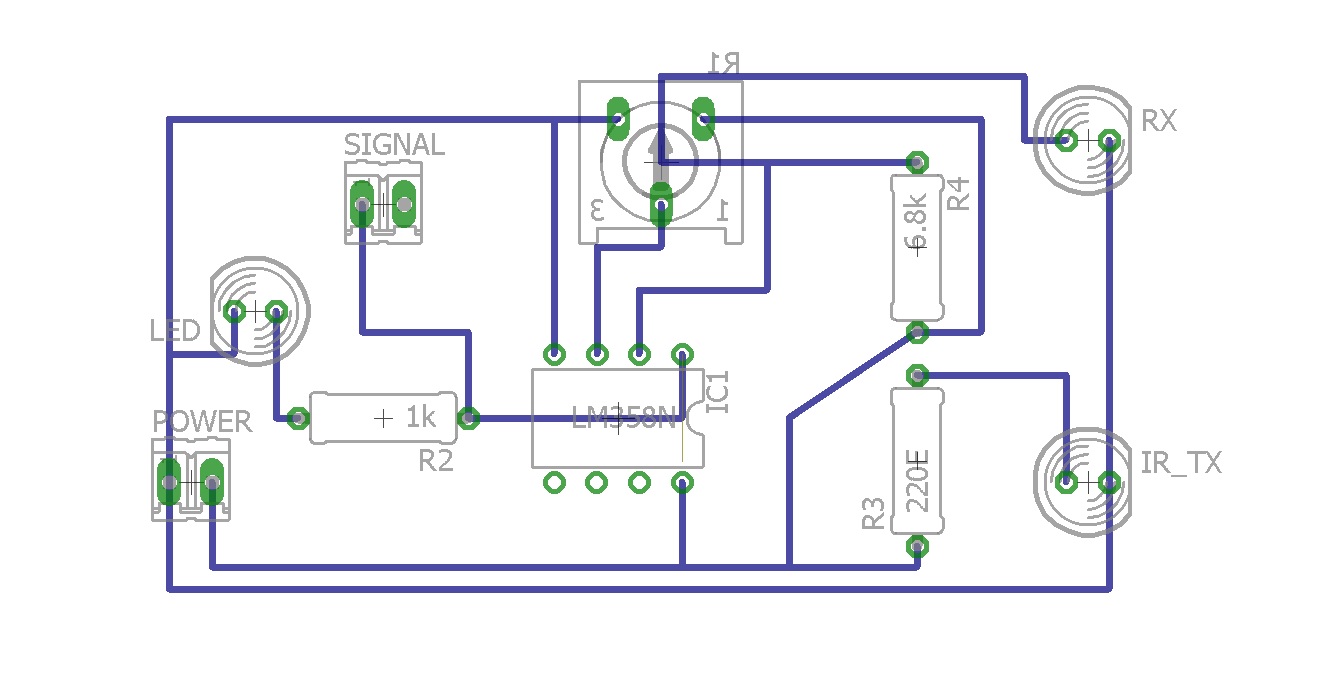


Fig 5.2.2 Board layout of circuit

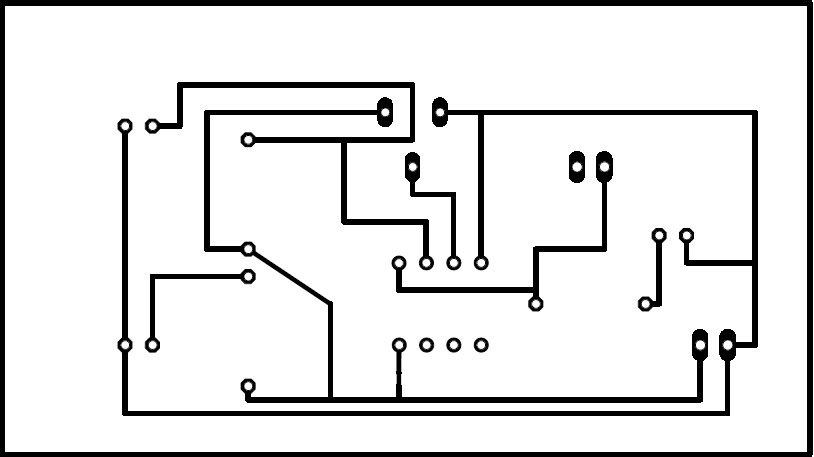


Fig 5.2.3 Track side PCB layout of circuit

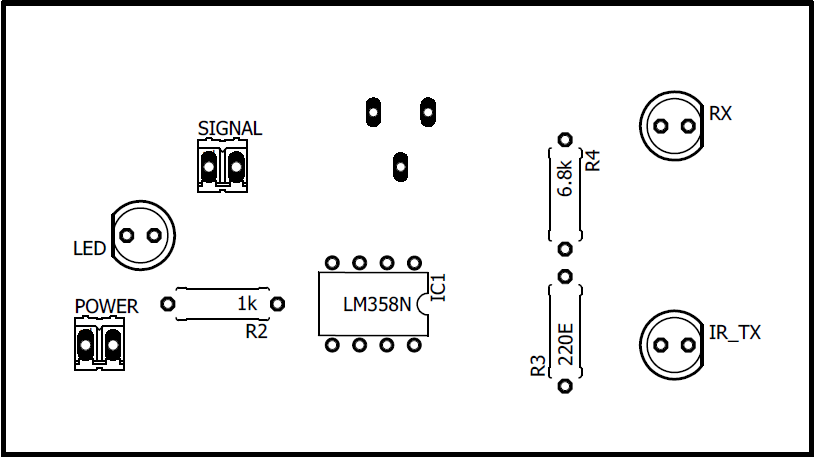


Fig 5.2.4 Component side PCB layout of circuit

**CHAPTER 6 RESULT**

|  |  |  |
| --- | --- | --- |
| Captured Image | Processed Image | Output text file |
|  |  |  |
|  |  |  |
|  |  |  |

Efficiency:

* In 12 out of 18 number plates, the program was able to isolate the number plate successfully.
* 8 out of 12 numbers plates had been identified successfully with error upto 2 characters only & the remaining number plates produced more than 2 errors.

**CHAPTER 7 CONCLUSION**

* We have successfully designed & implemented a portable ANPR system with small size factor & less power consumption using OpenCV, Python & Tesseract OCR engine.
* Designed ANPR system is connected to a Google Drive account for accessing the output images & number plate database.
* The ANPR that has been designed can be integrated with the exisiting infrastructure like CCTV at traffic signals & can thus be used for traffic monitoring and congestion charge billing .
* Furthermore, efficiency can be increased by using better OCR engine & IR illuminated camera.

**CHAPTER 8 BILL OF MATERIAL**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Component** | **Quantity** | **Cost(INR)** |
| 1 | Raspberry Pi | 1 | 3200/- |
| 2 | Webcam | 1 | 1000/- |
| 3 | LM 358 | 1 | 10/- |
| 4 | IR pair | 1 | 20/- |
| 5 | Connecting wires | 10 | 80/- |
| 6 | PCB | 1 | 150/- |
|  |  | **Total** | **4460/-** |

**CHAPTER 9 FUTURE SCOPE & APPLICATIONS**

The process of Automobile Number Plate Recognition requires a very high degree of accuracy when we are working on a very busy road or parking which may not be possible manually as a human being tends to get fatigued due to monotonous nature of the job and they cannot keep track of the vehicles when there are multiple vehicles are passing in a very short time. To overcome this problem, an effort can be made by the researchers across the globe by taking the systems image input using online hardware and specialized cameras installed at appropriate location where the automobile number plate recognition system is implemented. Though we have achieved an accuracy of 90-95% by optimizing various parameters, it is required that for the task as sensitive as tracking stolen vehicles and monitoring vehicles for homeland security an accuracy of 100% cannot be compromised with. Therefore, to achieve this, further optimization is required. Also, the issues like stains, smudges, blurred regions & different font style and sizes are need to be taken care of. This work can be further extended to minimize the errors due to image capturing.

* In electronic toll system
* Identify and record the license plate details of all vehicles coming into, and/or leaving, a particular area.
* Traffic monitoring and congestion charge billing
* Mass surveillance
* Generating billing information for parking

**CHAPTER 10 BIBLIOGRAPHY**

1. Raja Vikramdeep Singh, Navneet Randhawa, “Automobile Number Plate Recognition and Extraction Using Optical Character Recognition”, International Journal Of Scientific & Technology Research Volume 3, Issue 10, October 2014.
2. M. T. Qadri, Muhammad Asif of Sir Syed, “Automatic Number Plate Recognition System for Vehicle Identification Using Optical Character Recognition”, International Conference on Education Technology and Computer, 2009.
3. Er. Kavneet Kaur, Vijay Kumar Banga,” Number Plate Recognition Using OCR Technique”, IJRET: International Journal of Research in Engineering and Technology, Volume: 02 Issue: 09 | Sep-2013
4. Tesseract- ‘https://github.com/tesseract-ocr/tesseract’

**APPENDIX**