**CNN Model for Number Plate Detection**

A MINI PROJECT REPORT

Submitted in partial fulfilment of the Requirement for the award of the degree

of

**Bachelor of Engineering**

**In**

**Information Technology**

**By**

**N.Sushanth (1608-21-737-002)**

**G.Karthik (1608-21-737-025)**

**R.Venkata Anirudh (1608-21-737-054)**

**Under the guidance**

**of**

**Dr. J.Srinivas**

Associate Professor



**Department of Information Technology**

**MATRUSRI ENGINEERING COLLEGE**

**SAIDABAD – 500059**

**AUGUST 2024**

# MATRUSRI ENGINEERING COLLEGE

**SAIDABAD – 500059**



## Department of Information Technology

CERTIFICATE FROM PROJECT GUIDE

This is to certify that the project report entitled “**CNN Model For Number Plate Recognition**” submitted by **N.Sushanth** bearing H.T. No**: 1608-21-737-002, G.Karthik** bearing H.T. No**: 1608-21-737-025, R.venkata Anirudh** bearing H.T. No: **1608- 21-737-054**, in the partial fulfilment of the requirement for the award of the degree of Bachelor of Engineering in Information Technology is a Bonafide work carried by them. The results of the investigations enclosed in this report have been verified and found satisfactory.

**Dr.J.Srinivas Dr. G. Shyama ChandraPrasad**

**Associate professor HOD,**

**Dept of IT , Dept of IT,**

**MECS MECS**

# MATRUSRI ENGINEERING COLLEGE

**SAIDABAD – 500059**



## Department of Information Technology

### DECLARATION BY THE CANDIDATE

We **N.Sushanth** bearing H.T. No:1608-21-737-002, **G.Karthik** bearing H.T.No: 1608-21-737-025, **R,Venkata Anirudh** bearing H.T.No: 1608-21-737-054 hereby certify that the mini project report entitled “**CNN Model For Number Plate Recognition**” is submitted in the partial fulfilment of the required for the award of the degree of **Bachelor of Engineering in Information Technology.**

This is a Record of Bonafide work carried out by us under the guidance of **DR.J.Srinivas**, Associate Professor, Department of IT Matrusri Engineering College, Saidabad. The results embodied in this report have not been reproduced/copied from any source. The results embodied in this report have not been submitted to any other university or institute for the award of any other degree or diploma.

**N.Sushanth (1608- 21-737-002)**

**G.Karthik (1608-21-737-025)**

**R.Venkata Anirudh (1608-21-737-027)**

**Dept of IT, MECS**

## ACKNOWLEDGEMENT

We wish to take this opportunity to express our deep gratitude to all the people who have extended their cooperation in various ways during our mini project work. It’s our pleasure to acknowledge the help of all those individuals.

Firstly, we would like to thank **Dr. G. Shyama Chandra Prasad** (HOD, IT DEPT) for his encouragement and valuable guidance in bringing shape to dissertation.

We would like to thank our project guide**, Dr.J.Srinivas** Associaate professor, Department of IT for his guidance and help throughout the development of this project work by providing us with required information and support. Without his guidance, cooperation and encouragement, we couldn’t have learnt many new things during our mini project tenure.

**N.Sushanth (1608- 21-737-002)**

**G.Karthik (1608-21-737-025)**

**R.Venkata Annirudh (1608-21-737-054)**

**Dept of IT, MECS**

## ABSTRACT

The License Plate Detection System is an advanced automated application developed using Python and OpenCV. This project explores the implementation of a Convolutional Neural Network (CNN) for accurately detecting and recognizing vehicle number plates from images or video streams. Harnessing the robust capabilities of CNNs for image processing, the system aims to achieve high precision and efficiency in number plate recognition.

The preferred language for development is Python due to its comprehensive libraries and ease of use, though the concepts can be implemented in any language supporting CNN and OpenCV. The system employs image pre-processing techniques to enhance the input images, followed by the use of CNN for feature extraction and number plate detection.

Key steps include converting the image to grayscale, applying Gaussian blur, and using edge detection techniques to identify potential number plate regions. The CNN model is then used to classify these regions accurately. This method ensures that the system can handle various lighting conditions and image qualities effectively.

This number plate detection system offers a significant step towards automated vehicle identification, with potential applications in traffic management, toll collection, and security systems.

Keywords: License Plate Detection, CNN, OpenCV, Python, Image Processing.

### TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **CONTENTS** | **PAGE NO.** |
| 1 | Introduction | 8-13 |
| 2 | System specification | 14-18 |
| 3 | Literature Survey | 19-20 |
| 4 | Methodology | 21-22 |
| 5 | UML Diagrams | 23-28 |
| 6 | Sample code | 29-33 |
| 7 | Output | 34-36 |
| 8 | Conclusion | 37 |
| 9 | Future Enhancements | 37-38 |
| 10 | References | 38-39 |

### LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **FIGURE** | **PAGE NO.** |
| 1 | Architecture diagram | 11 |
| 2 | Data Flow Diagram | 13 |
| 3 | Class Diagram | 25 |
| 4 | Use Case Diagram | 26 |
| 5 | Activity Diagram | 27 |
| 6 | Sequence Diagram | 28 |

### 1. INTRODUCTION `

License plate recognition (LPR) has become an essential tool in various sectors, including law enforcement, traffic management, and automated toll collection. The evolution of image detection and recognition technologies has significantly advanced LPR systems. Initially, LPR systems relied on basic image processing techniques like edge detection and template matching, which were limited by their inability to handle variations in lighting, angle, and plate design. These early systems often struggled with accuracy and required substantial computational resources.

The advent of machine learning and deep learning, particularly Convolutional Neural Networks (CNNs), has revolutionized image detection. CNNs, with their ability to learn and extract features from images, have drastically improved the accuracy and robustness of LPR systems. These models can now handle diverse conditions, such as different lighting, angles, and different front styles making LPR more reliable and efficient.

Despite these advancements, there is a continuous need for improvement. Real-world applications demand even higher accuracy and faster processing speeds. Challenges such as recognizing dirty or damaged plates, dealing with high-speed vehicle motion, and ensuring privacy and security require ongoing innovation. Thus, the development of more sophisticated models and algorithms remains crucial to meet the ever-growing demands of modern LPR systems.

**1.1 Objective:**

The objective of this project is to develop a highly accurate and efficient license plate recognition system using Convolutional Neural Networks (CNNs) and OpenCV, capable of detecting and recognizing vehicle number plates under diverse conditions to enhance traffic management, law enforcement, and automated toll collection systems.

#### 1.2 Existing System

Currently, license plate recognition systems rely on traditional computer vision algorithms which may not always provide accurate results, especially in challenging conditions such as varying lighting, occlusion, different front styles and complex backgrounds. These systems often require manual tuning and are not always reliable.

**1.2.1 Disadvantages:**

1. Variable Accuracy

2. High Cost

3. Real-Time Processing Challenges

4. Environmental Impact

5. Limited Adaptability

6. Privacy Concerns

**1.3** **Proposed System:**

To enhance license plate recognition, we propose a solution using a Convolutional Neural Network (CNN) model. First, a diverse dataset of license plate images is collected, including various designs, lighting conditions, and angles. The images undergo pre-processing steps like augmentation, normalization, and noise reduction to improve data quality. The CNN architecture comprises convolutional layers for feature extraction, max pooling to reduce dimensionality, batch normalization to accelerate training, and dropout layers to prevent overfitting. For license plate detection, contours are used to identify potential plate regions, refined using bounding box regression. Once detected, characters within the plate are segmented and recognized using the CNN model. Post-processing steps include character validation, using rules or secondary models to correct errors, and cross-referencing recognized plates with a database for validation. The model is trained with a well-labeled dataset, employing appropriate loss functions and optimizers, and its performance is evaluated on a test dataset for accuracy and robustness. This CNN-based approach aims to improve recognition accuracy and efficiency, effectively handling image variations and complexities inherent in license plate recognition tasks.

**1.3.1 Advantages:**

* High Accuracy
* Robust Feature Extraction
* Improved Detection
* Scalability
* Real-Time Processing
* Versatility
* Noise Reduction
* Automated Character Segmentation
* Error Correction
* Continuous Improvement

#### 1.4 Modules

**1.4.1 General Architecture**

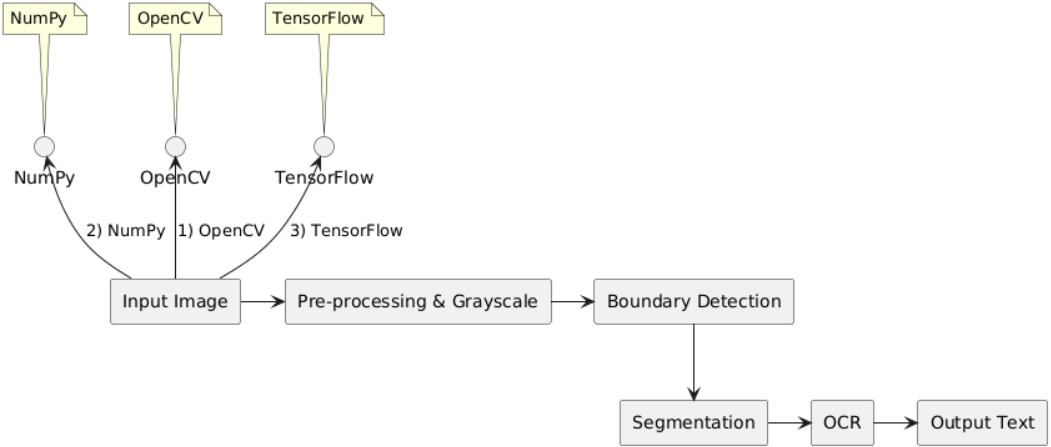


Figure 1.4.1. Architecture Diagram

The architecture diagram Fig 1.4.1 depicts a license plate recognition system, starting with the installation of essential libraries (OpenCV, NumPy, and TensorFlow). The process begins with input image pre-processing and grayscale conversion, followed by boundary detection and segmentation. OCR then reads the segmented characters, outputting the text. This setup enables real-time object detection and drawing, enhancing the software's frame reading and drawing capabilities.

##### 1.4.2. Design Phase :

The design phase of the license plate recognition system involves a detailed blueprint of the system architecture, modules, data flow, and necessary components.

The system architecture comprises several stages, each dedicated to a specific task within the license plate recognition process. The primary components include input image acquisition, pre-processing and grayscale conversion, boundary detection, segmentation, optical character recognition (OCR), and output text display. Each component plays a vital role in ensuring the system's accuracy and efficiency.

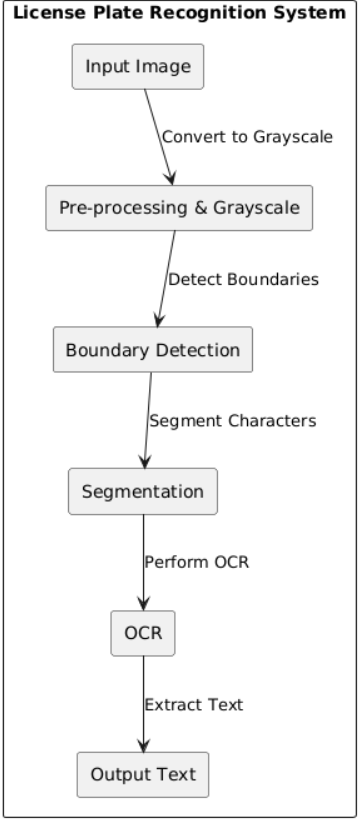
The input image acquisition module is responsible for capturing or uploading an image containing a license plate. OpenCV is utilized for image capture from a camera or loading from a file. Once the image is acquired, it is sent to the pre-processing and grayscale conversion module, where the image is converted to grayscale to simplify the subsequent processing steps. This conversion is performed using OpenCV.

The next stage involves boundary detection, where the edges or boundaries of the license plate within the image are identified. This is achieved using edge detection algorithms . The detected boundaries are then passed to the segmentation module, which segments the license plate region into individual characters. Contour detection and extraction methods in OpenCV facilitate this process.

Following segmentation, the segmented characters are sent to the OCR module. This module's purpose is to recognize and convert the characters into text using the Tesseract OCR library. The recognized text is then forwarded to the output text display module, which displays the extracted text (license plate number) to the user.

The data flow in the system starts with the input image being converted to grayscale in the pre-processing stage. The grayscale image undergoes boundary detection to identify the license plate's edges, followed by segmentation to isolate individual characters. The segmented characters are processed by the OCR module to extract text, which is then displayed as output.

**1.4.3. Data Flow Diagram**



The sequence of operations involves the user providing an image to the system, which then processes the image through the aforementioned stages. The recognized text is ultimately displayed to the user, completing the license plate recognition process.

### 2.SYSTEM SPECIFICATION

#### 2.1. Hardware Specification

* CPU: Intel Core i5 or Ryzen 7 or higher
* GPU (recommended for faster training): NVIDIA GTX 1060 or higher
* RAM: 8GB or higher
* Storage: 100GB or higher
* Input Devices: Keyboard, Mouse
* Display: Monitor (any size)

#### 2.2. Software Specification

**Operating system** – Windows 10 or higher

**Visual Studio code :**

Visual Studio Code is a code editor redefined an optimized for building and debugging modern web and cloud applications.

**Programming language:**

**Why Python?**

It has a huge number of libraries and frameworks. The Python language comes with many libraries and frameworks that make coding easy. This also saves a significant amount of time.

The most popular libraries are NumPy, which is used for scientific calculations; SciPy for more advanced computations; and scikit, for learning data mining and data analysis. These libraries work alongside powerful frameworks like TensorFlow, CNTK, and Apache Spark. These libraries and frameworks are essential when it comes to machine and deep learning projects.

**Simplicity:** Python code is concise and readable even to new developers, which is beneficial to machine and deep learning projects. Due to its simple syntax, the development of applications with Python is fast when compared to many programming languages. Furthermore, it allows the developer to test algorithms without implementing them.

**The massive online support**: Python is an open-source programming language and enjoys excellent support from many resources and quality documentation worldwide. It also has a large and active community of developers who provide their assistance at any stage of development. Most scientists have adopted Python for Machine Learning and Deep Learning projects, which means most of the brightest minds worldwide, can be found in Python communities.

**Fast development:** Python has a syntax that is easy to understand and friendly. Furthermore, the numerous frameworks and libraries boost software development. By using out-of-box solutions, a lot can be done with a few lines of code.

**Flexible integrations:** Python projects can be integrated with other systems coded in different programming languages. This means that it is much easier to blend it with other AI projects written in other languages. Also, since it is extensible and portable, Python can be used to perform cross languages tasks. The adaptability of Python makes it easy for data scientists and developers to train machine learning models.

**Fast code tests:** Python provides a lot of code review and test tools. Developers can quickly check the correctness and quality of the code.

AI projects tend to be time-consuming, so a well-structured environment for testing and checking for bugs is needed.

Python is the ideal language since it supports these features.

**Performance:** Some developers argue that Python is relatively slow compared to other programming languages. As much as speed is not one of Python’s strong suits, it provides the solution known as Cython. It is a superset of Python language designed to achieve code performance the same as C language. Developers can use Cython to code C extensions the same way they code in Python, as its syntax is almost the same. Cython increases the language performance significantly.

**Visualization tools:** Python comes with a wide variety of libraries. Some of these frameworks offer good visualization tools. In AI, Machine learning, and Deep learning, it is important to present data in a human-readable format. Therefore, Python is a perfect choice for implementing this feature.

Python is used in this project for its robust libraries and ease of use. It handles image acquisition and processing with OpenCV, performs numerical operations with NumPy, and handles segmentation and boundary detection tasks. Python's extensive library support and simplicity make it ideal for integrating these components seamlessly.

**Technologies used:**

**OpenCV:**



OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today’s systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

**Numpy:**



NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software.

It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code.

**TensorFlow:**



TensorFlow is a comprehensive open-source platform for machine learning. It provides a flexible and high-performance framework for building and deploying machine learning models. Developed by the Google Brain team, TensorFlow is widely used in research and production environments.

It contains various features including these important ones:

1. Easy model building: Use high-level APIs like Keras with eager execution for immediate iteration and debugging.
2. Robust ML production anywhere: Deploy machine learning models on any platform, including servers, mobile devices, and edge devices.
3. Powerful experimentation for research: Simplify the process of training, evaluating, and fine-tuning machine learning models.
4. Flexible and efficient: Supports CPU, GPU, and TPU computing, enabling efficient use of hardware resources.
5. Extensive ecosystem: Integrates with a variety of libraries and tools for tasks like data preprocessing, model evaluation, and visualization.

### 3. LITERATURE SURVEY

# [1]M.A. Jawale a, P. William a, A.B. Pawar b, Nikhil Marriwala c Implementation of number plate detection system for vehicle registration using IOT and recognition using CNN

In this paper, an Automatic License Plate Detection and Recognition (ALPDR) system has been proposed having four steps namely License Plate Extraction, Image Pre-processing, Character Segmentation and Character Recognition. For the first three steps (extraction, pre-processing, and segmentation), unique methods have been proposed. As the character recognition is an important step of license plate recognition and detection, four different methods for character recognition have been experimented on, which include Convolution Neural Network (CNN), MobileNet, Inception V3, ResNet

1. **Chen, M., AlRegib, G. and Juang, B.H., 2015. “Air-writing recognition—Part I: Modeling an-drecognition of characters, words, and connecting motions”. IEEE Transactions on HumanMachineSystems,46(3), pp.403-413.**

Air-Writing refers to writing of linguistic characters or words in a free space by hand or finger movements.Here recognition of characters or words is accomplished based on six-degree-of-freedom hand motion data. Isolated air-writing characters can be recognized similar to motion gestures although with increased sophistication and variability.

1. **Joolee, J.B., Raza, A., Abdullah, M. and Jeon, S., 2020. “Tracking of Flexible Brush Tip on-Real Canvas: Silhouette-Based and Deep**

**Ensemble Network-Based Approaches”. IEEE Access,8,pp.115778115788.**

We introduced silhouette-based and deep ensemble network-based approaches to track the brush tip position for interactive drawing. The silhouette-based approach captures the silhouette of deforming bristles using a pair of well-aligned infra-red (IR) cameras, extracts the tip using our proposed tracking procedure and then the 2D position of the tip is reconstructed. However, this approach still needs a specially aligned frame and cameras and has shortcoming in usability. In the current work, we only consider a standard size brush.

1. **Kaur, H., Reddy, B.G.S., Sai, G.C. and Raj, A.S., 2021. “A Comprehensive overview of AR/VR by Writing in Air”.**

In this experiment dependency injection OpenCV is used to sketch on the camera with a virtual pen,i.e. any marker may be used to draw using the contour detection technique centered on the maskof the desired cultured reference marker. The research is all about how often people could identify alphabets and numbers written in the open air. A leap motion captures motion trajectory information and plots it as a continuous stream of points. Lines would be combined and major slopes identified from the major points. Significant slopes are converted into directions by the use of geometry.

### 4. METHODOLOGY

The methodology for the license plate recognition system involves a series of stages, each dedicated to specific tasks to achieve accurate and efficient recognition of license plates from images. The process begins with image acquisition, where images of vehicles are captured or loaded. This step is facilitated by the OpenCV library, which provides powerful tools for handling and manipulating images.

Once the image is acquired, the next stage involves pre-processing and grayscale conversion. This step simplifies the image by reducing it to shades of gray, which helps in highlighting the important features required for further processing. OpenCV is again utilized here to perform the grayscale conversion and other necessary pre-processing tasks such as noise reduction and normalization.

Following pre-processing, the system moves to the boundary detection stage. In this phase, the system identifies the edges or boundaries of the license plate within the grayscale image. This is achieved using edge detection algorithms like the Canny edge detector, which is implemented through OpenCV. The purpose of this stage is to isolate the license plate region from the rest of the image, making it easier to focus on relevant details.

The isolated license plate region is then subjected to segmentation. Segmentation involves breaking down the license plate region into individual characters. This is done using contour detection and extraction techniques provided by OpenCV. By isolating each character, the system prepares the data for the final recognition phase.

For the character recognition phase, a machine learning or deep learning model is used. These models are trained to recognize and classify characters from segmented images. Libraries such as TensorFlow or PyTorch can be employed to build and deploy these models. The model processes the segmented character images and outputs the corresponding text.

Finally, the recognized text is gathered and can be output in various formats depending on the system requirements. The output can be displayed on the screen, stored in a database, or used in further processing systems such as automated toll collection or traffic monitoring systems.

Overall, Python's extensive libraries and straightforward syntax facilitate each stage of the process, making it an ideal choice for developing an efficient and robust license plate recognition system.

### 5.UML DIAGRAMS

* UML is a graphical notation used to visualize, specify, construct and document the artifact of software intensive. UML is appropriate for modelling systems ranging from Enterprise Information Systems to Distributed Web-based Application and even to Hard Real-time Embedded systems. UML effectively starts with forming a conceptual modelling of the language. UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. OMG is continuously making efforts to create a truly industry standard.
* UML Standard for Unified Modelling Language.
* UML is different from the other common programming languages such as C++, Java, COBOL, etc.
* UML is a pictorial language used to make software blueprints.
* UML can be described as a general-purpose visual modelling language to visualize, specify, construct, and document software system.

Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard.

UML diagrams are not only made for developers but also for business users, common people, and anybody interested to understand the system. The system can be a software or non-software system. Thus, it must be clear that UML is not a development method rather it accompanies with processes to make it a successful system. In conclusion, the goal of UML can be defined as a simple modelling mechanism to model all possible practical systems in today’s complex environment.

#### 5.1 Class diagram

A class is a set of objects that share a common structure and common behavior (the same attributes, operations and semantics). A class is an abstraction of real-world things. When these items exist in the realworld, they are instances of the class and are referred to as objects.

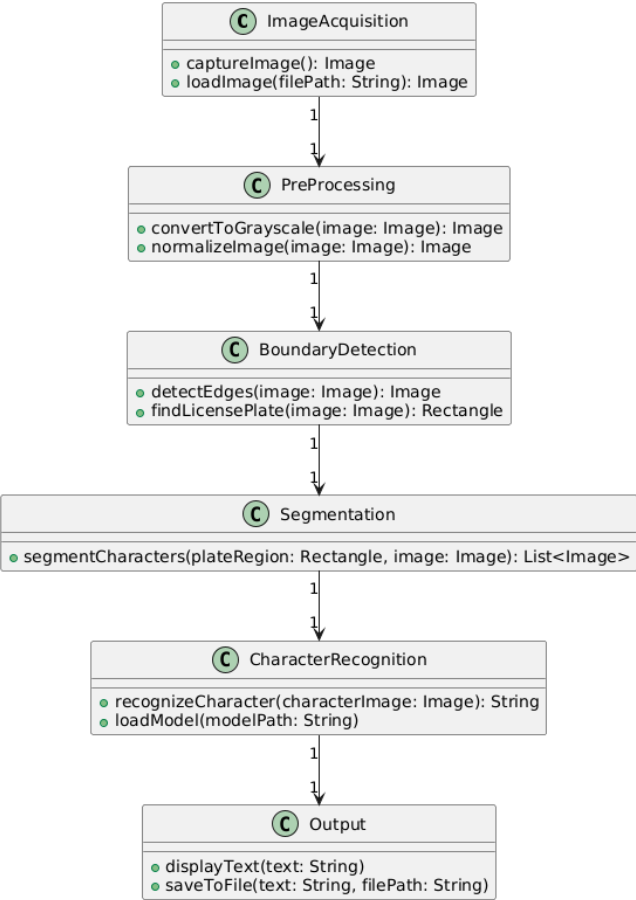


Figure 5.1 Class diagram

The figure 4.1. is a static diagram. The class diagram illustrates the flow from image acquisition through pre-processing, boundary detection, segmentation, and character recognition, culminating in output display or file saving. Each stage is represented by a dedicated class.

#### 5.2 Use case diagram

A use case diagram describes a set of sequences in which each sequence indicates the relation with outside things. A use case involves the interaction of actor and system.

In below figure 4.2. describes a high level functions and scope of system. It also shows the interaction between the system and the users. Here the user acts like actor and python IDE acts like a system.

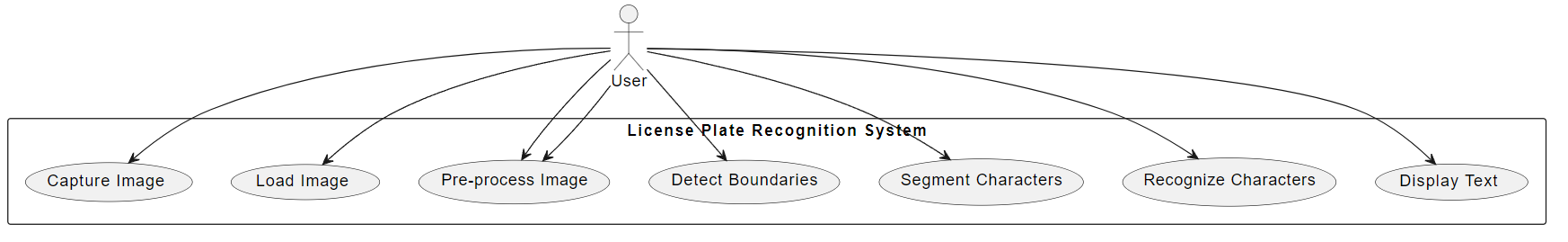
****

Figure 5.2. Usecase Diagram

#### 5.3 Activity Diagram

An activity diagram is a special case of state diagram. An activity diagram is like a flow Machine showing the flow a control from one activity to another. An activity diagram is used to model dynamic aspects of the system.

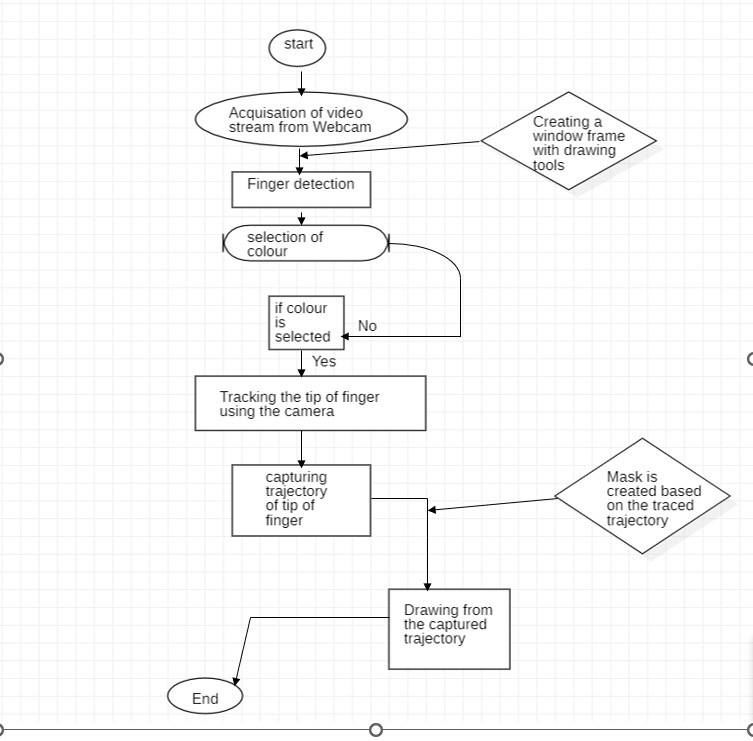


Figure 5.3 Activity Diagram

#### .4 Sequence Diagram

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence, what happens first, what happen next.

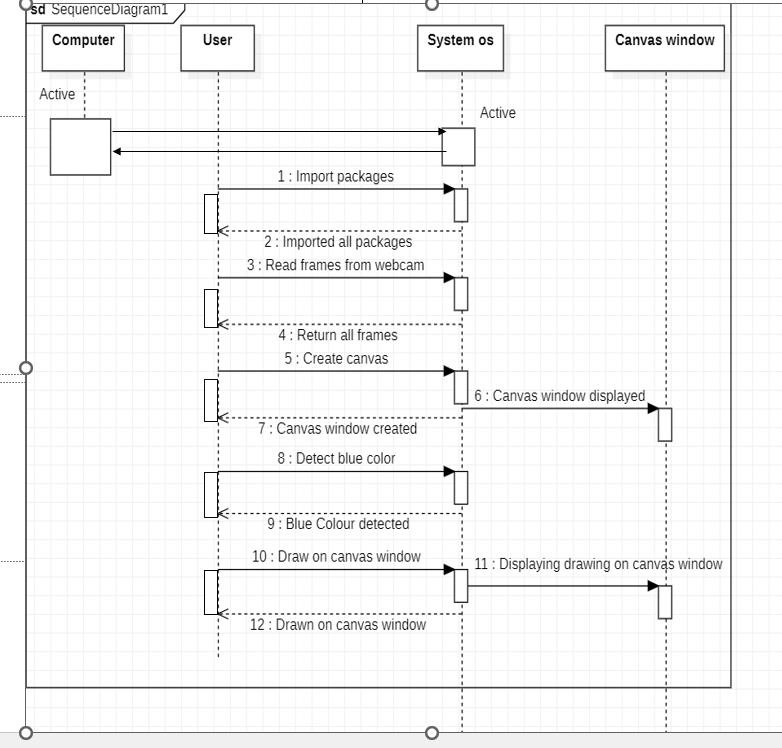


Figure 5.4 Sequence Diagram

In the figure 4.6 we are initially checking whether computer and system OS are in active state or not. After checking the user have to install the necessary packages. Now read frames from the webcam and create the canvas window. Check whether canvas window is created or not. After creation of window, detect the blue color from webcam. Now return or draw the frames on the canvaswindow and display the canvas window on screen.

### 6. SAMPLE CODE

im

### 7.OUTPUT SCREEN

#### 7.1. Canvas window creation

Figure 7.1. Canvas window creation

#### 7.2. Object Detection

Figure 7.2. Object Detection

The user uses a blue color object which was detected by camera and the it reads the frames and drawn on the paint window.

#### 7.3. Drawing on canvas

Figure 7.3. Drawing on Canvas

The output will be displayed on paint window. The resultant output is based on the input given by the user.

### 8. CONCLUSION

Day by day technology is improving a lot. In our present technology we want to draw our imaginations by just waving our finger in the air. Thus we got a new technology of building an Air canvas with which can draw anything on the canvas by just capturing the motion of a colored marker with a laptop camera. It will again work towards a greater purpose in helping especially those who know them to communicate easily. Even adults who find it difficult to use the keyboard can easily use the program.

### 9. FUTURE ENHANCEMENTS

To enhance hand gesture tracking, we would have to delve more into Open CV. There are many different methods of contour analysis, but in this particular algorithm, it may be worthwhile to take a look at the color histogram used to create the contours in question. Whenever a new green object comes into the frame it takes the new object a pencil. This will be improved so that the using object remains as the pencil even if a high intensity new object enters the frame. In future a tools tab will be added which contains mathematical tools like triangle,rectangle etc for easy drawing. In future additional features will be added so that light colors will be drawn when we selected a particular color.

### 10.REFERENCES

1. Babu, S., Pragathi, B.S., Chinthala, U. and Maheshwaram, S., 2020, September. Subject Tracking with Camera Movement Using Single Board Computer. In 2020 IEEE-HYDCON (pp. 1-6). IEEE.
2. Chen, M., AlRegib, G. and Juang, B.H., 2015. Air-writing recognition—Part I: Modeling and recognition of characters, words, and connecting motions. IEEE Transactions on Human-Machine Systems, 46(3), pp.403-413.
3. Joolee, J.B., Raza, A., Abdullah, M. and Jeon, S., 2020. Tracking of

Flexible Brush Tip on Real Canvas: Silhouette-Based and Deep Ensemble Network-Based Approaches. IEEE Access,8, pp.115778115788.

1. Kaur, H., Reddy, B.G.S., Sai, G.C. and Raj, A.S., 2021. A Comprehensive overview of AR/VR by Writing in Air.

1. Shetty, M., Daniel, C.A., Bhatkar, M.K. and Lopes, O.P., 2020, June. Virtual Mouse Using Object Tracking. In 2020 5th International Conference on Communication and Electronics Systems(ICCES) (pp. 548-553). IEEE.

1. Zhou, L., 2019. Paper Dreams: an adaptive drawing canvas supported by machine learning (Doctoral dissertation, Massachusetts Institute of Technology).

1. Erik B. Sudderth, Michael I. Mandel, William T. Freeman, Alan S. Willsky, “Visual Hand Tracking Using Nonparametric Belief Propagation”, Mit Laboratory For Information Decision Systems Technical Report P-2603, Presented at IEEE CVPR Workshop On Generative Model Based Vision, Pp. 1-9, 2004.

1. Yuan-Hsiang Chang, Chen-Ming Chang, “Automatic Hand-Pose Trajectory Tracking System Using Video Sequences”, INTECH, pp. 132-

152, Croatia, 2010

1. Yusuke Araga, Makoto Shirabayashi, Keishi Kaida, Hiroomi Hikawa,

“Real Time Gesture Recognition System Using Posture Classifier and Jordan Recurrent Neural Network”, IEEE World Congress on

Computational Intelligence, Brisbane, Australia, 2012

1. Ruiduo Yang, Sudeep Sarkar, “Coupled grouping and matching for sign and gesture recognition”, Computer Vision and Image Understanding, Elsevier, 2008