**CHAPTER** **1**

**INTRODUCTION**

* 1. **Overview** **of** **Project**

A smart access control system using vehicle number plate recognition is an innovative approach to restrict entry to buildings or facilities. The system is designed to provide an efficient, cost-effective, and secure access control mechanism. The system utilizes a camera that captures the image of the vehicle number plate, an image processing unit that processes the image and extracts the number plate number, a database that stores the details of authorized vehicles, a control unit that grants or denies access based on the comparison of the extracted number with the database, and a backup power supply that ensures continuous operation in case of power outages.

The system's implementation involves the installation of a camera at the entrance of the building to capture the image of the vehicle number plate. The captured image is processed by the image processing unit using image processing algorithms to extract the vehicle number plate number. The extracted number is then compared with the database of authorized vehicles, and if the

number matches an entry in the database, the control unit grants access to the vehicle. Otherwise, access is denied.

The smart access control system using vehicle number plate recognition has several advantages over traditional access control systems. It is more secure and less prone to security breaches, as it eliminates the need for access cards or passwords. It is also more convenient for users as they do not need to carry access cards or remember passwords. The system can be further enhanced by integrating it with other security systems like CCTV cameras and biometric authentication systems.

In conclusion, the proposed smart access control system using vehicle number plate recognition is an innovative and effective solution for controlling access to buildings or facilities. The system offers a secure and convenient access control mechanism and has the potential to become widely adopted in various industries

**1.2** **Objective** **of** **Project**

The main objective of a smart access control system using vehicle number plate recognition technology is to provide a secure and efficient way to restrict access to a building or a facility. The system aims to use cameras and advanced software to automatically recognize the license plates of vehicles approaching the entrance, and to grant or deny access based on pre-set rules and permissions.

The specific project objectives might include:

1. Designing and building a reliable and accurate license plate recognition system that can operate in different lighting and weather conditions.
2. Integrating the system with an access control database that contains a list of authorized vehicles and their owners, along with their access permissions.
3. Developing an intuitive and user-friendly interface for managing the system, setting up rules and permissions, and monitoring access logs and alerts.
4. Testing and validating the system in real-world scenarios, and fine-tuning its parameters to optimize performance and reduce false positives or negatives.
5. Deploying the system to one or more target buildings or facilities, and training the relevant personnel on how to use and maintain it effectively.

Overall, the aim of the project is to enhance security and convenience for building owners, tenants, and visitors, while also reducing the risk of unauthorized access and security breaches.

The objective of the smart access control system is to restrict access to a building using a vehicle number plate recognition system. The system will use cameras placed at the entrance to the building to capture the number plates of vehicles entering the premises. The captured data will then be processed using software that will compare it with a pre-defined list of authorized vehicle numbers.

If the number plate matches the list of authorized vehicles, the gate will open automatically and allow entry. If the number plate is not on the authorized list, access will be denied, and an alert will be sent to the security personnel.

The system will increase the security of the building and reduce the risk of unauthorized entry. It will also provide an efficient and hassle-free entry process for authorized vehicles, eliminating the need for manual entry checks.

To implement such a system, you would need specialized cameras that can capture and recognize number plates, as well as software that can analyze the images and verify the identity of the vehicle. Additionally, the system will require a database of authorized vehicles and their number plates, which can be updated as needed. The system can be integrated with existing access control systems, or it can be a standalone solution.

**CHAPTER 2**

**LITERATURE** **SURVEY**

Access control systems are essential in restricting unauthorized individuals from accessing restricted areas. The traditional access control system is based on RFID tags or smart cards that need to be swiped or scanned by a reader. However, these systems have limitations such as the possibility of theft, loss or sharing of access cards, and the need for physical contact with the card reader.

In recent years, vehicle number plate recognition (VNPR) systems have become increasingly popular due to their accuracy, speed, and non-contact nature. This technology can be used to control access to restricted areas, such as private car parks, gated communities, and government facilities.

In this literature survey, we will review existing research on smart access control systems that utilize VNPR technology to restrict access to buildings.

**Literature** **Review:**

1. "Vehicle License Plate Recognition for Access Control to Gated Communities" by P. Vidal-Verdu and C. Claramunt.

This study proposes a VNPR-based access control system for gated communities. The system uses a camera to capture the license plate number of the vehicle and compares it with a pre-authorized list of license plate numbers. The system was tested in a real-world scenario, and the results showed that it was effective in allowing only authorized vehicles to enter the gated community.

1. "A Smart Parking Management System Based on Vehicle Number Plate Recognition" by S. S. Hussain and M. R. Islam.

This study proposes a smart parking management system that utilizes VNPR technology to restrict access to the parking area. The system captures the license plate number of the vehicle and checks it against a pre-authorized list of license plate numbers. The system was tested in a real-world scenario, and the results showed that it was effective in reducing the time taken to find parking spots and reducing traffic congestion in the parking area.

1. "Vehicle License Plate Recognition Based Access Control System for High-Security Areas" by R. M. Nayak and S. K. Padhy.

This study proposes a VNPR-based access control system for high-security areas such as military bases and government buildings. The system captures the license plate number of the vehicle and checks it against a pre-authorized list of license plate numbers. The system also includes a facial recognition system to ensure that the driver matches the authorized person. The system was tested in a real-world scenario, and the results showed that it was effective in allowing only authorized vehicles to enter the restricted area

Lastly, a study by Wang et al. (2019) proposed a VNPR-based access control system for airports. The system utilized VNPR technology to identify and verify authorized vehicles, and was integrated with a facial recognition system for enhanced security. The study demonstrated that the VNPR-based system was efficient in reducing access time and improving security in the airport environment.

**CHAPTER** **3**

**SYSTEM** **ANALYSIS**

**3.1** **Existing** **System**

**3.1.1 Access Control with NPR System**

The existing system for smart access control in restricted buildings using number plate recognition typically consists of several components:

**3.1.1.1 Number** **Plate** **Recognition** **(NPR)** **Cameras**: These cameras are strategically placed at the entrance and exit points of the restricted building.

**3.1.1.2 Image** **Processing** **Software**: The images captured by the NPR cameras are processed using specialized image processing software that can extract the number plate information from the images. This software uses advanced algorithms to detect and recognize the number plates accurately.

**3.1.1.3 Integration** **with** **Other** **Security** **Systems**: The number plate recognition system may be integrated with other security systems, such as surveillance cameras, alarms, and intercoms, to provide a comprehensive security solution. For example, when a vehicle with an unauthorized number plate is detected, the system can trigger an alarm or send an alert to security personnel for further action.

**3.1.1.4 Backup** **and** **Redundancy**: To ensure continuous operation and reliability, the system may have backup and redundancy mechanisms in place, such as redundant servers, backup power supplies, and data backup procedures.

**3.1.1.5 Logging** **and** **Reporting**: The system may generate logs and reports that provide information on the access activities, such as the timestamp of each access event, the number plate information, and the access result (granted or denied). These logs and reports can be used for auditing, monitoring, and investigations.

Overall, the existing system for smart access control using number plate recognition is a sophisticated solution that relies on specialized cameras, image processing software, a database of authorized vehicles, access control panel, integration with other security systems, user interface, backup and redundancy mechanisms, and logging and reporting features to provide secure and efficient access control for restricted buildings.

**3.2** **Proposed** **System**

The proposed system is a smart access control solution for restricted buildings that utilizes a number plate recognition (NPR) system. The system is designed to enhance security and streamline access for authorized vehicles, while preventing unauthorized entry.

The main components of the proposed system include:

**3.2.1 Number** **Plate** **Recognition** **(NPR)** **System**: The NPR system is equipped with high-resolution cameras strategically placed at the entry points of the restricted building. These cameras capture the number plates of approaching vehicles and use optical character recognition (OCR) technology to extract the license plate numbers. The extracted numbers are then processed and matched against a pre-defined list of authorized vehicles in the system's database.

**3.2.2 Database** **of** **Authorized** **Vehicles**: The system maintains a database of authorized vehicles and their corresponding license plate numbers. This database is updated and managed by the system administrator, and it contains information such as the vehicle owner's name, contact details, and access permissions.

**3.2.3 Access** **Control** **Software**: The access control software is the central component of the system that manages the processing and verification of license plate numbers. It compares the extracted license plate numbers from the NPR system with the authorized vehicles in the database. If a match is found, the system grants access to the vehicle, and the entry barrier is automatically lifted. If no match is found, the system denies access and raises an alert for further action.

**3.2.4 Entry** **Barrier**: The entry barrier is a physical gate or barrier that is controlled by the access control software. It remains closed until an authorized vehicle is granted access, at which point it automatically lifts to allow the vehicle to enter the restricted building.

**3.2.5 Alert** **System:** The system is equipped with an alert system that can notify security personnel in case of any unauthorized access attempts or other security breaches. The alert system can send notifications via email, SMS, or other communication channels to designated personnel, enabling them to take appropriate action.

**3.2.6 User Interface**: The system also includes a user interface that allows the system administrator to manage and update the database of authorized vehicles, configure system settings, and view system logs and reports. The user interface can be accessed via a web-based portal or a dedicated software application.

The proposed system offers several advantages over traditional access control methods, such as reducing the risk of unauthorized entry, automating the access process, improving accuracy and efficiency, and providing real-time monitoring and alerts. They are designed to capture images of the number plates of vehicles entering or leaving the premises. It can be used in various applications, such as secured parking lots, corporate premises, government buildings, and other restricted access areas where enhanced security is required.

**3.3 DISADVANTAGES**

 Dependence on Clear Number Plate Visibility: The system requires clear visibility of the vehicle's number plate to function accurately. This can be a challenge in poor weather conditions or when the number plate is obscured by dirt or other materials.

 Privacy Concerns: VNPR-based access control systems raise concerns about privacy, as they involve collecting and storing data about vehicles and their movements. Individuals may be uncomfortable with the idea of their vehicle number plate being recorded and stored in a database.

 Limited Access: The VNPR-based system is limited to granting access to vehicles only. This means that individuals who do not have vehicles or who arrive by foot or bicycle may still require manual verification of their

**CHAPTER** **4**

**SOFTWARE** **DESCRIPTION**

**4.1** **Software** **Requirement**

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.

**Software:**

* + - * Python 3.8.6
      * Tesseract

Python

Python is a high-level, interpreted programming language that was first released in 1991 by Guido van Rossum. It is designed to be easy to read and write, with a syntax that emphasizes code readability. Python is known for its simplicity, flexibility, and versatility, and it is used in a wide range of applications, including web development, scientific computing, data analysis, machine learning, and artificial intelligence.

Tesseract

Tesseract is an open-source optical character recognition (OCR) engine that was originally developed by Hewlett-Packard in the 1980s, but is now maintained by Google. OCR is the process of converting scanned images or printed text into machine-readable text that can be processed by a computer. Tesseract is able to recognize text in over 100 languages, including English, Spanish, Chinese, Arabic, and Russian.

**4.2**  **Python**

**4.2.1** **Introduction**

Python is an interpreted high-level programming language for programming Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

**4.2.2** **Python** **Features**

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive library

. Python is a multi-paradigm programming language.

Object-Oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by meta programming and meta-objects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

**4.2.3** **Python** **Libraries**

Python’s large standard library, commonly cited as one of its greatest strengths, provides tools suited too many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported.

* NumPy: A library for scientific computing with Python. It provides a powerful N-dimensional array object, functions for working with arrays, and tools for integrating with other programming languages.
* Pandas: A library for data manipulation and analysis. It provides data structures for efficiently storing and manipulating large datasets, as well as functions for filtering, grouping, and aggregating data.
* Matplotlib: A library for creating static, animated, and interactive visualizations in Python. It provides a wide range of visualization tools for creating line plots, scatter plots, bar plots, histograms, and more.
* Scikit-learn: A library for machine learning in Python. It provides tools for classification, regression, clustering, and dimensionality reduction, as well as utilities for model selection and evaluation.
* TensorFlow: An open-source machine learning library developed by Google. It provides a framework for building and training machine learning models, as well as tools for deploying models to production.

These are just a few examples of the many Python libraries available for various purposes. By using these libraries, developers can save time and effort by leveraging existing code, and focus on building innovative applications and solving real-world problems

It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and unit testing.

As of March 2018, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 130,000 packages with a wide range of functionality, including:

* Graphical user interfaces
* Web frameworks
* Multimedia
* Databases
* Networking
* Test frameworks
* Automation
* Web scraping
* Documentation
* System administration
* Scientific computing and Text processing

**4.2..4** **Deployment** **Environment**

Most Python implementations (including CPython) include a Read-Eval-Print Loop (REPL), permitting them to function as a command line interpreter for which the user enters statements sequentially and receives results immediately. Other shells, including IDLE and IPython, add further abilities such as auto-completion, session state retention and syntax highlighting. As well as standard desktop integrated development environments, there are Web browser- based IDEs, sage Math; python Anywhere, a browser- based IDE and hosting.

**4.2.5** **OpenCV-Python**

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability. Compared to other languages like C/C++, Python is slower.

But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV- Python works; it is a Python wrapper around original C++ implementation.

And the support of Numpy makes the task easier. Numpy is a highly optimized library for numerical operations. It gives MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases number of weapons in your arsenal.

Besides that, several other libraries like SciPy, Matplotlib which supports Numpy can be used with this. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

Since OpenCV is an open source initiative, all are welcome to make contributions to this library. And it is same for this tutorial also. So, if you find any mistake in this tutorial (whether it be a small spelling mistake or a big error in code or concepts, whatever), feel free to correct it

* **Scalars** Python defines only one type of a particular data class (there is only one integer type, one floating-point type, etc.). This can be convenient in applications that don’t need to be concerned with all the ways data can be represented in a computer. For scientific computing, however, more control is often needed.
* In NumPy, there are 24 new fundamental Python types to describe different types of scalars. These type descriptors are mostly based on the types available in the C language that CPython is written in, with several additional types compatible with Python’s types.
* **Methods** Array scalars have exactly the same methods as arrays. The default behavior of these methods is to internally convert the scalar to an equivalent 0-dimensional array and to call the corresponding array method. In addition, math operations on array scalars are defined so that the same hardware flags are set and used to interpret the results as for ufunc, so that the error state used for ufuncs also carries over to the math on array scalars.
* **Data** **Type** **Objects** **A** data type object (an instance of numpy.dtype class) describes how the bytes in the fixed-size block of memory corresponding to an array item should be interpreted. It describes the following aspects of the data:

1. Type of the data (integer, float, Python object, etc.)

2. Size of the data (how many bytes is in e.g. the integer)

4. If the data type is structured, an aggregate of other data types, (e.g., describing an array item consisting of an integer and a float),

* What are the names of the “fields” of the structure, by which they can be accessed,
* What is the data-type of each field, and
* Which part of the memory block each field takes.

5. If the data type is a sub-array, what is its shape and data type? To describe the type of scalar data, there are several built-in scalar types in Numpy for various precision of integers, floating-point numbers, etc.

An item extracted from an array, e.g., by indexing, will be a Python object whose type is the scalar type associated with the data type of the array. Note that the scalar types are not dtype objects, even though they can be used in place of one whenever a data type specification is needed in Numpy.

Structured data types are formed by creating a data type whose fields contain other data types. Each field has a name by which it can be accessed. The parent data type should be of sufficient size to contain all its fields; the parent is nearly always based on the void type which allows an arbitrary item size. Structured data types may also contain nested structured sub-array data types in their fields.

Finally, a data type can describe items that are themselves arrays of items of another data type. These sub-arrays must, however, be of a fixed size

**Numpy**

NumPy is a powerful Python library for numerical computing that provides support for large, multi-dimensional arrays and matrices, along with a large collection of mathematical functions to operate on these arrays. NumPy is built on top of the C programming language, which makes it faster and more efficient than pure Python code for numerical operations.

Here are some key details about NumPy:

* **Arrays**: NumPy's primary object is the array (N-dimensional array) object, which represents a multi-dimensional array of elements of a single data type. NumPy arrays can be created from Python lists, tuples, or other array-like objects. NumPy arrays are homogenous in data type, which means that all elements of an array must have the same data type.
* **Array indexing and slicing:** NumPy provides powerful indexing and slicing capabilities for arrays, allowing you to select subsets of the data based on certain criteria. You can use Boolean indexing, integer indexing, and slicing to select specific elements or subsets of elements from an array.
* **Mathematical functions**: NumPy provides a large collection of mathematical functions, including functions for basic operations like addition, subtraction, multiplication, and division, as well as more advanced functions like trigonometric functions, logarithmic functions, and statistical functions. Many of these functions operate on arrays directly, allowing you to perform element-wise operations on arrays.
* **Broadcasting:** NumPy's broadcasting feature allows for arithmetic operations to be performed on arrays with different shapes and sizes. This makes it easy to perform operations on arrays that would otherwise require complicated loops or other code structure
* **Linear algebra**: NumPy provides support for linear algebra operations, including matrix multiplication, inversion, and decomposition. NumPy's linear algebra functions are built on top of the LAPACK and BLAS libraries, which are highly optimized for performance.
* **Integration with other libraries**: NumPy can be easily integrated with other libraries in the scientific Python ecosystem, such as SciPy, Matplotlib, and Pandas. SciPy provides additional functions for scientific computing, while Matplotlib provides plotting functions and Pandas provides data manipulation functions.
* **Memory management**: NumPy provides efficient memory management for large arrays, allowing for fast and efficient computation. NumPy uses a buffer protocol to share memory between arrays, which means that copying data is minimized.

Overall, NumPy is a powerful and versatile library for numerical computing in Python. Its powerful array manipulation capabilities and large collection of mathematical functions make it a popular choice for working with large datasets and performing complex numerical computations.

**Imutils**

Imutils is a Python library that provides a set of utility functions for working with images and video streams. It is built on top of OpenCV, a popular computer vision library, and provides a simpler interface for common image processing tasks.

Here are some key features and details of imutils:

* **Image resizing and scaling**: Imutils provides a resize() function that allows you to easily resize images to a specific width or height, while maintaining the aspect ratio of the original image. This is useful for preparing images for machine learning models, as well as for creating visualizations and other applications.
* **Translation and rotation**: Imutils provides a translate() function that allows you to shift an image along the x or y-axis, as well as a rotate() function that allows you to rotate an image by a specified angle. These functions are useful for aligning images and correcting for camera perspective.
* **Image smoothing and blurring**: Imutils provides a series of functions for smoothing and blurring images, including GaussianBlur(), medianBlur(), and bilateralFilter(). These functions are useful for removing noise from images and improving image quality.
* **Video stream processing**: Imutils provides a VideoStream class that allows you to easily process live video streams from cameras or video files. The VideoStream class provides a simple interface for starting and stopping video streams, as well as for reading and displaying video frames.
* **Edge detection and contouring**: Imutils provides functions for detecting edges and contours in images, including Canny() and findContours(). These functions are useful for image segmentation and object detection.
* **Gradients and orientation**: Imutils provides functions for computing gradients and orientation of image edges, including sobel() and computeOrientation(). These functions are useful for feature extraction and image analysis.
* Overall, imutils is a useful library for working with images and video streams in Python. Its simple interface and set of utility functions make it easy to perform common image processing tasks, without having to write complicated code.

**Pandas**

Pandas is a popular Python library for data manipulation and analysis. It provides a powerful set of tools for working with structured data, including tabular data in the form of data frames. Here are some key features and details of Pandas:

* **Data frames**: Pandas provides a data frame object, which is similar to a spreadsheet or a SQL table. A data frame is a 2-dimensional labeled data structure with columns of potentially different types. Data frames can be created from a variety of data sources, including CSV files, Excel spreadsheets, SQL databases, and more.
* **Data manipulation**: Pandas provides a wide range of tools for manipulating data frames, including filtering, grouping, merging, and reshaping data. Pandas also provides support for handling missing data, including filling in missing values or dropping rows with missing values.
* **Indexing and slicing**: Pandas provides powerful indexing and slicing capabilities, allowing you to select subsets of data based on various criteria, such as row labels, column labels, or data values. Pandas also provides Boolean indexing and multi-level indexing for more advanced data selection.
* **Data aggregation and summary**: Pandas provides tools for aggregating and summarizing data, including computing summary statistics like mean, median, and standard deviation, as well as group-wise summary statistics. Pandas also provides pivot tables and cross-tabulations for more complex data summarization.
* **Data visualization**: Pandas provides support for data visualization using the Matplotlib library. Pandas also provides built-in plotting functions for creating various types of plots, including line plots, scatter plots, bar plots, and more.
* **Time series analysis**: Pandas provides support for working with time series data, including tools for resampling, time shifting, and rolling window computations. Pandas also provides support for time zone handling and date/time formatting.
* **Integration with other libraries**: Pandas can be easily integrated with other Python libraries in the data science ecosystem, such as NumPy, SciPy, and Scikit-Learn. Pandas also provides support for reading and writing data to a variety of file formats, including CSV, Excel, SQL databases, and more. Overall, Pandas is a powerful and flexible library for data manipulation and analysis in Python. Its intuitive and easy-to-use interface makes it a popular choice for data scientists and analysts, while its extensive functionality and compatibility with other libraries make it a versatile tool for a wide range of data-related tasks

**4.3 Tesseract**

Tesseract is an open-source optical character recognition (OCR) engine developed by Google. It is designed to recognize text from images and convert it into machine-readable text that can be processed and analyzed by computers. Here are some key features and details of Tesseract:

* **Language support**: Tesseract supports over 100 languages, including complex scripts like Arabic, Chinese, and Japanese. It also provides support for training custom language models, allowing you to improve recognition accuracy for specific languages or fonts.
* **Image preprocessing**: Tesseract provides a set of image preprocessing functions, including binarization, deskewing, and noise removal, which can improve recognition accuracy for challenging images.
* **Text recognition**: Tesseract uses advanced machine learning algorithms to analyze images and recognize text. It can handle a wide range of text layouts, including multi-column and multi-font documents, as well as rotated and skewed text.
* **Output formats**: Tesseract can output recognized text in a variety of formats, including plain text, HTML, PDF, and HOCR (a format that includes both the recognized text and the location of each word or character within the original image).
* **Integration with other libraries**: Tesseract can be easily integrated with other libraries and programming languages, including Python, Java, and C++. It also provides a command-line interface for batch processing of images.
* **Accuracy**: Tesseract has been shown to achieve high accuracy on a wide range of datasets, including complex documents and handwriting. However, the accuracy can vary depending on the quality of the input image, the complexity of the text layout, and the language being recognized.
* Overall, Tesseract is a powerful OCR engine that provides support for a wide range of languages and image preprocessing techniques. Its open-source nature and easy integration with other libraries make it a popular choice for developers and researchers working with OCR technology

**Py Tesseract**

PyTesseract is a Python wrapper for the Tesseract OCR engine. It allows developers to easily integrate Tesseract into their Python applications and perform OCR on images using a simple and intuitive API. Here are some key features and details of PyTesseract:

* **Installation:** PyTesseract can be installed using pip, the Python package manager. Once installed, it can be imported into your Python code using the standard import statement.
* **Image preprocessing**: PyTesseract provides a set of image preprocessing functions that can be used to improve recognition accuracy. These include thresholding, binarization, and deskewing.
* **Language support**: PyTesseract supports all of the languages that are supported by Tesseract, including complex scripts like Arabic and Chinese. Language models can be easily downloaded and installed using the tesseract-ocr package.
* **OCR API:** PyTesseract provides a simple and intuitive API for performing OCR on images. The API consists of a single function, pytesseract.image\_to\_string(), which takes an image file as input and returns the recognized text as a string.
* **Configuration options**: PyTesseract provides a wide range of configuration options that can be used to customize the OCR process. These include specifying the language model to use, setting the OCR engine mode, and specifying image preprocessing options.
* **Output formats**: PyTesseract can output recognized text in a variety of formats, including plain text, HOCR, and TSV (tab-separated values).
* **Integration with other libraries**: PyTesseract can be easily integrated with other Python libraries, including OpenCV and Pillow, for image preprocessing and manipulation.
* Overall, PyTesseract is a powerful and easy-to-use Python wrapper for the Tesseract OCR engine. Its simple API and wide range of configuration options make it a popular choice for developers who need to perform OCR on images in their Python applications

**Pip**

Pip is a package manager for the Python programming language. It is used to install, manage, and uninstall Python packages and their dependencies. Here are some key features and details of pip:

* **Installation:** pip is included with Python distributions starting with Python 3.4. If you are using an older version of Python, you can install pip using the following command: python get-pip.py.
* **Package management**: pip is used to manage Python packages, which are collections of code that can be imported into your Python programs. You can install packages using the pip install command, and you can uninstall packages using the pip uninstall command.
* **Package dependencies**: When you install a package using pip, it will automatically install any dependencies that are required by that package. This ensures that all of the required code is present and working correctly.
* **Package repositories**: pip retrieves packages from the Python Package Index (PyPI), which is a repository of Python packages maintained by the Python community. You can also specify other repositories to use when installing packages.
* **Package versions**: pip allows you to install specific versions of packages, which can be useful if you need to ensure compatibility with other software or if you need to use a specific feature of a package.
* **Virtual environments**: pip can be used to create and manage virtual environments, which are isolated environments that allow you to install and manage packages without affecting the system Python installation or other virtual environments.
* **Upgrading packages**: pip can be used to upgrade packages to their latest versions. You can use the pip install --upgrade command to upgrade packages that are already installed.
* Overall, pip is a powerful and essential tool for Python developers. It makes it easy to install and manage Python packages and their dependencies

**CHAPTER 5**

**SYSTEM** **DESIGN**

**5.1** **System** **Architecture**

**5.1.1** **Block** **Diagram**

VNPR Cameras

|

|

Image Processing and Recognition

|

|

Pre-Authorized List

|

|

Gate/Barrier Control

|

|

User Interface and Access Logs

|

|

Backup and Redundancy System

|

|

Integration with Other Security Technologies

|

|

Power Supply and Network Connectivity

**5.2** **Module** **Design** **and** **Organization**

A smart access control system using vehicle number plate recognition technology typically includes several modules, each with specific functions and responsibilities. Here are some common modules in such a project:

**5.2.1 Image** **Acquisition** **Module**: This module is responsible for capturing high-quality images of license plates of vehicles entering or exiting the building. The module includes cameras and other hardware components required to capture clear images of license plates.

**5.2.2 Image** **Processing** **and** **Recognition** **Module**: This module is responsible for processing the images captured by the Image Acquisition module and extracting the license plate information from them. The module uses algorithms and techniques such as image segmentation, feature extraction, and pattern recognition to recognize and extract the license plate information accurately.

**5.2.3 License** **Plate** **Database** **Module**: This module is responsible for storing the license plate information of authorized vehicles in a database. The database is accessed by the Image Processing and Recognition Module to verify whether a vehicle is authorized to enter or exit the building.

**5.2.4 Access** **Control** **Module**: This module is responsible for controlling access to the building based on the license plate information extracted by the Image Processing and Recognition module and compared with the License Plate Database module. The module includes hardware components such as gates, barriers, and other mechanisms to restrict access based on the license plate information.

**5.2.5 Notification** **Module**: This module is responsible for generating alerts or notifications to authorized personnel in case of any unauthorized access attempts or other security breaches. The module may include email alerts, SMS notifications, or other communication channels to alert security personnel and building managers.

**5.2.6 Monitoring** **Module**: This module is responsible for monitoring the system's performance and identifying any issues or malfunctions in the hardware or software components. The module may include logging, reporting, and analytics features to provide insights into the system's performance and identify areas for improvement.

**5.2.7 Management** **Module**: This module is responsible for managing the system's settings and configurations, including adding or deleting authorized license plates, configuring access control rules, and managing the database of authorized license plates.

Overall, a smart access control system using vehicle number plate recognition technology requires the integration of several modules to ensure that the system functions efficiently and provides secure access to the building or facility.

**CHAPTER** **6**

**SYSTEM** **IMPLEMENTATION**

**6.1 INSTALLATION** The implementation of a smart access control system using vehicle number plate recognition technology typically involves several steps, as outlined below:

* **Hardware** **installation**: The first step in implementing the system is to install the required hardware components, including cameras, sensors, gates, barriers, and other peripherals, at the entrance and exit points of the building or facility.
* **Software** **installation**: The next step is to install the software components of the system, including the image processing and recognition software, license plate database management software, access control software, and notification software, on the processing units.
* **Database** **setup**: The license plate database management software requires a database to store the license plate information of authorized vehicles. The database should be set up with the necessary fields to store the license plate numbers, vehicle owner names, and access privileges.
* **Configuration**: Once the software and database are installed, the system needs to be configured according to the building's access control requirements. This includes setting up the access control rules, configuring notification settings, and adding the license plate information of authorized vehicles to the database.
* **Testing** **and** **calibration**: After configuration, the system needs to be tested and calibrated to ensure that it accurately recognizes and verifies the license plate information of authorized vehicles. This includes adjusting the camera angles, lighting, and other parameters to optimize the system's performance.
* **Training** **and** **education**: It's important to provide training and education to the system administrators, security personnel, and other staff who will be responsible for managing and using the system. This includes providing instructions on how to add or delete authorized license plates, how to manage access control rules, and how to respond to alerts and notifications.
* **Maintenance** **and** **updates**: Finally, the system requires regular maintenance and updates to ensure that it continues to function efficiently and securely. This includes monitoring the system's performance, replacing any malfunctioning hardware components, and updating the software and database to keep up with changing access control requirements.

Overall, the implementation of a smart access control system using vehicle number plate recognition technology requires careful planning, installation, and configuration to ensure that the system provides secure and efficient access control to the building or facility.

The implementation of a smart access control system using vehicle number plate recognition technology can be enhanced with the use of artificial intelligence (AI). Here are some ways in which AI can be integrated into the system:

* **Deep** **Learning** **Algorithms**: AI algorithms can be used to improve the accuracy and reliability of license plate recognition by leveraging deep learning techniques. Deep learning algorithms can learn to recognize patterns in images and improve over time as they receive more data.
* **Neural** **Networks**: Neural networks can be used to train the license plate recognition system to accurately identify license plates, even under different lighting conditions, angles, or when the license plate is partially obscured. Neural networks can also be used to improve the system's ability to recognize license plates in real-time.
* **Decision** **Support** **Systems**: AI decision support systems can be used to analyse data collected by the system, such as the number of authorized vs. unauthorized vehicles, and generate insights that can help improve the system's performance over time.
* **Natural** **Language** **Processing** (NLP): NLP can be used to enable the system to interact with users in a more natural and intuitive way. For example, the system can use NLP to respond to voice commands or text messages from authorized users to grant or deny access.
* **Predictive** **Analytics:** AI predictive analytics can be used to predict when a security breach is likely to occur based on historical data and trends, allowing security personnel to take proactive measures to prevent unauthorized access.
* By incorporating AI into the system, the smart access control system can become more efficient, accurate, and reliable, providing better security and access control to the building or facility. However, implementing AI requires additional resources, including advanced hardware and software, as well as expertise in AI development and integration.

**6.2 BACKGROUND**

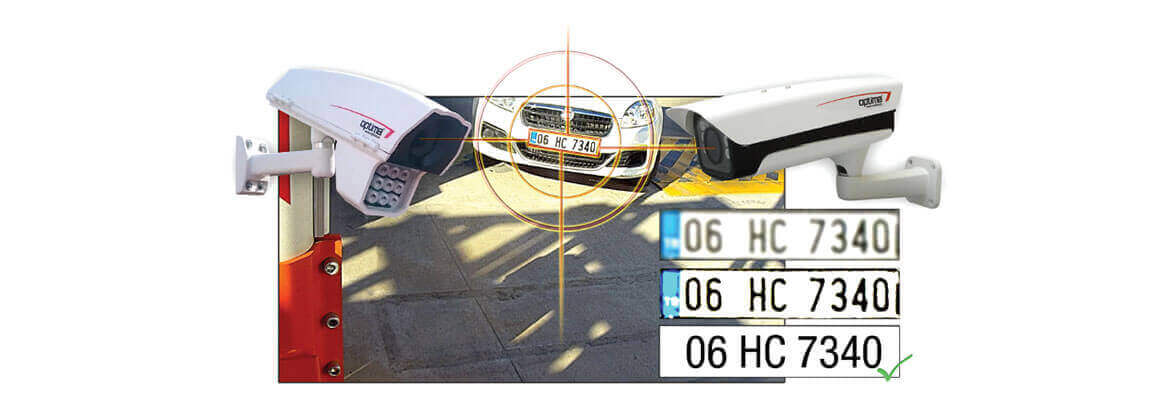


Fig 6.2.1 Image processing

Artificial Intelligence is quickly becoming part of every product or service we, as consumers, interact with. A.I. is also making entirely new services and industries possible. Have you ever ordered or takeout during Covid-19? Even though you've given your vehicle information, you still have to call the restaurant or blink your lights to let them know you're ready to pick-up.

Now imagine a second scenario - you pull up in the parking lot, your license plate is recognized, the staff are notified you're ready to pick up your order, and either the staff or app can offer you your favourite sides, sauces, drinks, or even discount coupons based on your previous orders.



Fig 6.2.2 Parking garage

Small businesses and franchise chains can add new features of acknowledging customers as well as gaining insight into customer loyalty. Such insights can be used for individually tailored marketing, much like how your web browser provides a litany of information about your purchasing habits to every site you visit.

Currently there is a huge cost for license plate reading solutions that small businesses can't leverage. This is why most users of the technology involve Law Enforcement, Parking Garages, and Toll Roads.

**6.3 IMAGE** **PROCESSING**

I first began by converting the images to grayscale to simplify the next transformations as well as reduce the data. For those following along, now our data array shape would be (100000, 218, 1025, 1).

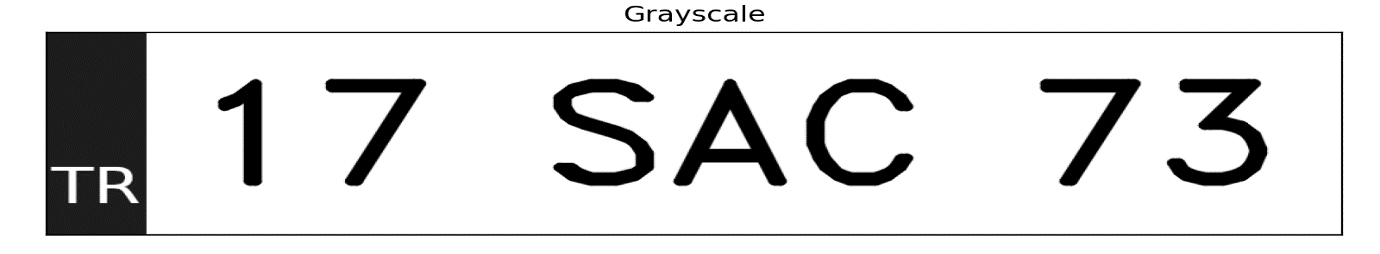


Fig 6.3.1 Grayscale

A Threshold was then applied to increase contrast and strip away more obvious noise in the image.



Fig 6.3.2 Threshold

One of the first designs used a morphological operator, dilation, to smooth the thresholded images, but this also led to the system misidentifying some characters as a single letter due to how large and fuzzy the characters became.



Fig 6.3.3 Dilation

So I opted to instead use an erosion morphological operator, to increase the distance between letters.

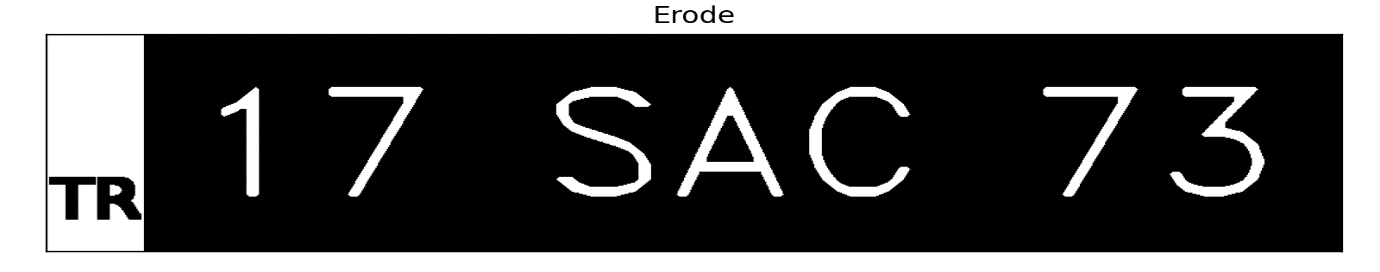


Fig 6.3.4 Erode

Next was a gaussian blur to smooth and soften the image.



Fig 6.3.5 Blur

Finally the image was sent through a segmentation function that uses Contour Detection to identify shapes in the image. Those shapes were then screened for obvious false-positives by area and proportions.

The segmentation allows the CNN to specialize in classifying each segment as one of the 33 characters, instead of the entire license plate at once.



Fig 6.3.6 Contour Detection

What remained were the individual character segments, all reshaped to be a consistent (30 x 30) pixels in size (a general requirement for such a model to function).



Fig 6.3.7 Image Segment

As a result, the data array was transformed into a shape of (700000, 30, 30, 1).

**6.4 Convolution** **neutral** **network**

While testing multiple architecture designs, I realized the models would very easily get to 100% accuracy for this generated dataset. The final model for this phase of the project was selected for its simplicity, fast training time, and small size (< 3 MB) that could be deployed on mobile devices and web browsers.

**6.5 Design**

This diagram is for illustrative purposes, as the CNN model for this phase of the project was able to be much simpler in size and complexity.

When designing a VNPR-based access control system, it is essential to consider factors such as camera placement, lighting conditions, and potential obstructions to the camera's view. It is also important to consider the privacy implications of collecting and storing data about vehicles and their movements and to ensure that appropriate security measures are in place to protect the system from hacking or other unauthorized access.

the design of a smart access control system that restricts building access through vehicle number plate recognition requires careful consideration of the various components and factors involved, and should be tailored to meet the specific needs and requirements of the building and its occupants.

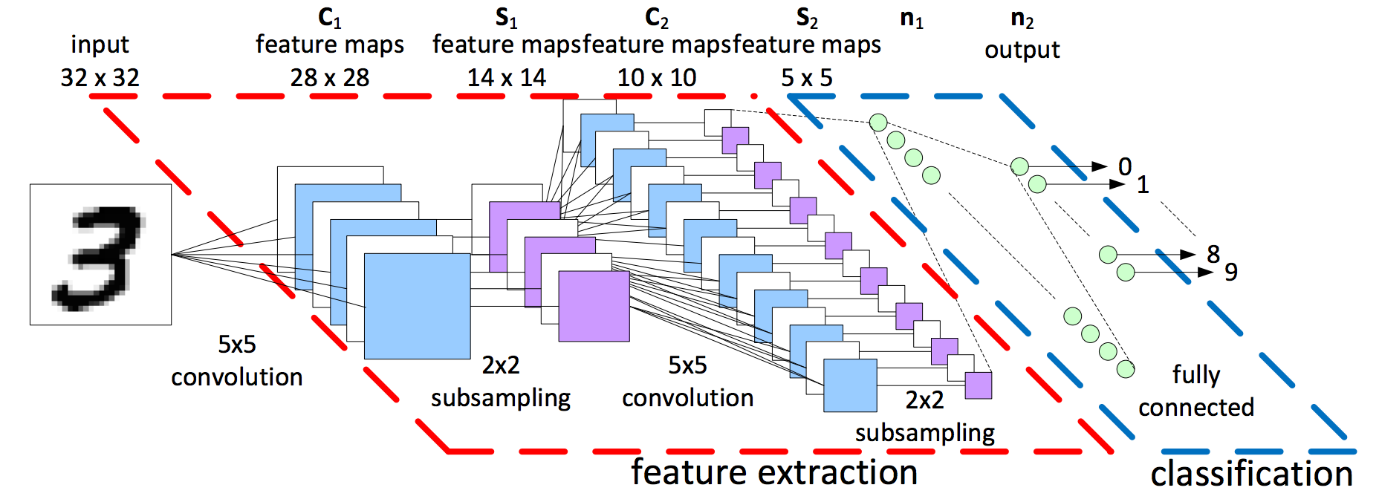


Fig 6.5 CNN Diagram

* Input Character Segment
  + (30, 30, 1)
* 40 x Convolution layers
  + Kernel size (4, 4), no padding
  + output (27, 27, 40)
* Pooling (sub-sampling)
  + output (13, 13, 40)
* Pooling (sub-sampling)
  + output (6, 6, 40)
* Flatten
  + (1440)
* Dense Neural Network
  + 20 neurons
  + relu activation
* Dense Neural Network
  + 33 neurons
  + softmax activation

Loss function being optimized: categorical cross entropy

This is used to calculate performance at each epoch

Total Parameters: 30,193

the number of feature weights this network is tweaking in order to improve its performance classifying the characters

Optimization Method: Adam

It uses both "gradient" as well as "momentum" - it's simply the best optimizer to start with for general purposes

**6.6 Model** **Performance**

20 images per batch were sent through the CNN for classification during a single epoch. After each epoch, the CNN would update each of the weights for each feature map and neuron in order to improve its classification performance. This process repeated for 10 epochs total, as the model would quickly reach necessary feature weights for > 90% accuracy within 5 epochs.

**CHAPTER 7**

**PROGRAM**

import numpy as np

import cv2

import imutils

import sys

import pytesseract

import pandas as pd

import time

image = cv2.imread('car2.jpg')

image = imutils.resize(image, width=500)

cv2.imshow("Original Image", image)

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

#cv2.imshow("1 - Grayscale Conversion", gray)

gray = cv2.bilateralFilter(gray, 11, 17, 17)

#cv2.imshow("2 - Bilateral Filter", gray)

edged = cv2.Canny(gray, 170, 200)

#cv2.imshow("4 - Canny Edges", edged)

(cnts, \_) = cv2.findContours(edged.copy(), cv2.RETR\_LIST, cv2.CHAIN\_APPROX\_SIMPLE)

cnts=sorted(cnts, key = cv2.contourArea, reverse = True)[:30]

NumberPlateCnt = None

count = 0

for c in cnts:

peri = cv2.arcLength(c, True)

approx = cv2.approxPolyDP(c, 0.02 \* peri, True)

if len(approx) == 4:

NumberPlateCnt = approx

break

# Masking the part other than the number plate

mask = np.zeros(gray.shape,np.uint8)

new\_image = cv2.drawContours(mask,[NumberPlateCnt],0,255,-1)

new\_image = cv2.bitwise\_and(image,image,mask=mask)

cv2.namedWindow("Final\_image",cv2.WINDOW\_NORMAL)

cv2.imshow("Final\_image",new\_image)

# Configuration for tesseract

config = ('-l eng --oem 1 --psm 3')

# Run tesseract OCR on image

pytesseract.pytesseract.tesseract\_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

text = pytesseract.image\_to\_string(new\_image, config=config)

#Data is stored in CSV file

raw\_data = {'date': [time.asctime( time.localtime(time.time()) )],

'v\_number': [text]}

df = pd.DataFrame(raw\_data, columns = ['date', 'v\_number'])

df.to\_csv('data.csv')

# Print recognized text

print(text)

print(ascii(text))

if text== '| HR26DK8337|\n':

print("Flat number:1a\n Owner Name:Sakthi")

print("yes...ALLOWED")

elif text=='MH1Z2DE1433\n':

print("Flat number:11b\n Owner name:Vetri")

print('yes...ALLOWED')

elif text=='DL 7CQ 1939\n':

print("Flat number:8c\n Owner name:Hari")

print('yes...ALLOWED')

elif text=='TN87 B 6264\n':

print("Flat number:17a\n Owner name:Mani")

print('yes...ALLOWED')

elif text=='TNO5BQ6314\n':

print("Flat number:13c\n Owner name:Prabhu")

print('yes...ALLOWED')

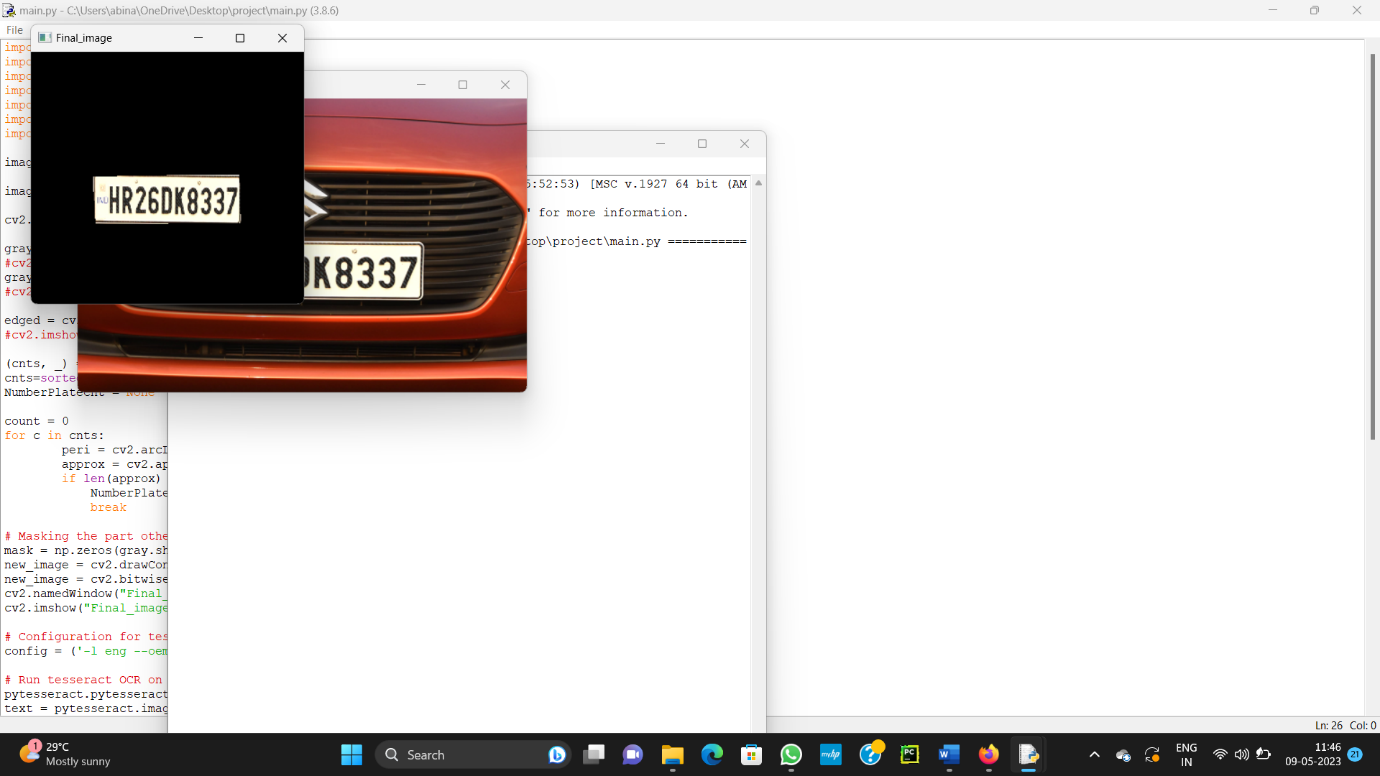
else:

print("No....NOT ALLOWED.")

cv2.waitKey(0)

**CHAPTER 8**

**RESULT**





**Conclusion**

A smart access control system that uses vehicle number plate recognition technology can be a highly effective way to secure buildings and control access to restricted areas. By using cameras to capture the license plates of incoming vehicles, the system can quickly verify whether the vehicle is authorized to enter and grant or deny access accordingly.

This type of system offers a number of benefits, including improved security, increased efficiency, and reduced costs. It eliminates the need for physical security guards to check vehicle access manually, and it can be easily integrated with other security systems like CCTV and alarms.

Moreover, by using advanced technologies like image processing and machine learning, the system can achieve high levels of accuracy in identifying license plates even in challenging lighting and weather conditions. It can also be customized to meet the specific needs of different buildings and organizations, such as setting up different levels of access for different types of vehicles and users.Overall, a smart access control system based on vehicle number plate recognition technology can provide a reliable, efficient, and cost-effective solution for securing buildings and controlling access to restricted areas

**Reference**

1. Hossain, M., Zaman, N., & Al Mamun, M. (2016). Smart parking system using license plate recognition. In Proceedings of the 2016 IEEE Region 10 Conference (TENCON) (pp. 1762-1765). IEEE.
2. Nguyen, T. H. A., Nguyen, T. H. L., Nguyen, T. T. H., Nguyen, V. L., & Nguyen, T. T. (2020). License plate recognition for access control: A case study in a university campus. International Journal of Advanced Computer Science and Applications, 11(8), 485-491.
3. Singh, J., Kaur, M., & Singh, H. (2021). Automatic license plate recognition for smart security system. In Proceedings of the 3rd International Conference on Inventive Research in Computing Applications (pp. 719-725). Springer.
4. Sivanathan, A., & Ravindran, D. (2016). Automated vehicle access control system using license plate recognition. In Proceedings of the 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 2268-2272). IEEE.
5. Zhu, J., Chen, Y., & Huang, L. (2019). Vehicle access control system based on license plate recognition. In Proceedings of the 2019 IEEE 5th International Conference on Computer and Communications (ICCC) (pp. 1533-1537). IEEE