**CHAPTER ONE**

**INTRODUCTION**

**1.1 Background of the Study**

Vehicles are an important part of today's transportation networks. A vehicle's license plate serves as a standard way of identification. An automatic license plate recognition system was created for this reason. Nowadays, recognizing or identifying the information present on moving objects is becoming of great importance in the world. The increasing population and the growth of industries has made the use of vehicles a mandatory need, which in turn leads in serious traffic issues on roads. The number of automobiles on the road is steadily increasing. Vehicle verification is not done in a systematic manner on the streets. As a result, on today's highways, the traditional approach of vehicle inspection is nearly impossible (Dhanlakshmi and Leni, 2017).

Automatic license/number plate recognition is a type of optical character recognition that is frequently used by law enforcement. Since its introduction, the use of automatic license plate recognition has expanded.   Automatic license plate recognition can be used to cite those who break traffic laws or drive too fast, as an electronic toll collecting system, to locate a suspect at a crime scene, or to identify uninsured drivers (when combined with a database search).

Plate readers have long been a law enforcement tool. They're used by cops to track down stolen vehicles. Poor file resolution is mainly caused by the plate being too far away from the camera, however it can also be caused by using a low-quality camera. While some of these issues can be resolved through software, the majority of these issues must be resolved through the hardware side of the system. With the use of image processing technology, the mobile-based scanning application system will simply scan the number plate; the resulting number will be utilized to obtain the vehicle's licensing information as well as the driver's profile. These facts, as well as the location, can be shared with our friends and family in the event of an emergency.

Due of plate mismatch from region to region, this technology favours specific places in terms of location. On photographs, ALPR employs visual character perception or Optical Character Recognition (OCR). Dutch car registration changed the font in 2002, adding small gaps to some letters (such as S and Q) to make them more specific and hence more lucid to such systems. Because different license plates have varying font sizes and placements, Automatic License Plate Recognition (ALPR) systems must be able to cope with such differences in order to be genuinely effective. Even though many algorithms are tailored to each country, more complex systems can withstand universal versions. Existing road-rule case or closed-circuit television cameras, as well as mobile groups that are generally connected to vehicles, can be used. Some systems use infrared cameras to capture a sharp image of the license plates. ALPR was discovered in 1976 at the Police Scientific Development Branch in the United Kingdom (Chowdhury et al., 2019).

Background subtraction and image colour segmentation both have a substantial impact on object detection, affecting the performance of locating probable items in the scene (Zhuang, Low and Yau, 2012). Background removal, inter-frame differencing, and optical flow are the three basic algorithms utilized in moving-object recognition in computer vision applications (Lian, Zhang and Liu, 2010). ALPR is also known as Automatic Vehicle Identification (AVI), Car Plate Recognition (CPR), Automatic Number Plate Recognition (ANPR), and Optical Character Recognition (OCR) for automobiles (Du, Ibrahim, Shehata and Badawy, 2013).

Image capture, de-blurring image frames, extracting license plate images, extracting characters from license plate images, recognizing license plate characters, and detecting cars are all part of the License Plate Recognition algorithm (Koval et al., 2003). The majority of existing ALPR systems process license plates using C++, but a few uses Matrix Laboratory. Furthermore, numerous other algorithms for detecting and recognizing license plates have been developed. Some systems necessitate the use of high-resolution cameras. Many researchers are concentrating their efforts these days on building computer vision-based automatic recognition systems. OpenCV is a free and open-source computer vision library that works with a variety of operating systems (OS) including Windows, Android, and Linux, as well as Python, Java, and C/C++. Tesseract is an open-source OCR engine that can run on a variety of operating systems. Before sending photos to Tesseract for OCR, they must be pre-processed (Buhuss, Timis and Apatean, 2016).

**1.2 Statement of the Research Problem**

Vehicle license plate recognition system adopted by the Nigerian Federal Road Safety Commission poses the following challenges:

1. Delaying road users.
2. Checking the innocent, and skipping the culpable.
3. Poor record keeping.

**1.3 Objective of the Study**

The primary aim of this work is to develop an IoT based vehicle license plate recognition system. The specific objectives are:

1. To develop a license plate detection module.
2. To integrate an optical character recognition engine.
3. To develop a real-time database management system.

**1.4 Significance of the Study**

The benefit of the vehicle license plate recognition and verification system is to make it easy for federal road safety officials to flag down only offenders without disturbing and delaying innocent road users and to reduce the risk of robbery and corruption and car accidents.

**1.5 Scope and Limitations of the Study**

This research work centres on design and implementation of an automatic real-time license plate recognition system that will be deployed using cameras. The limitation of this study includes the following:

1. This system does not recognize vehicle plates from several miles away.
2. The system does not confirm the availability of the vehicle accessories.
3. System functionality depends on the availability and strength of internet connection.

**1.6 Definition of Key Concepts**

1. **Automatic License Plate Recognition (ALPR):** Is the fundamental technology that locates a car license plate and then sends that information to a computer for further processing, where it can be processed, saved, or matched to construct an ALPR-based application.
2. **Deep Learning (DL):** Is a subset of artificial intelligence machine learning that uses networks to learn unsupervised from unstructured or unlabelled data. It can be called deep neural learning or deep neural network.
3. **You Only Look Once (YOLO):** Is one of the most powerful real-time object detector algorithms. It is called that way because unlike previous object detector algorithms, like Region-Based Convolutional Neural Network (R-CNN) or its upgrade Faster R-CNN, it only needs the image or video to pass one time through its network.
4. **Google Cloud Platform (GCP):** Is a real-time object detecting method that is one of the most powerful. Unlike prior object detecting algorithms such as Region-Based Convolutional Neural Network (R-CNN) or its upgrade Faster R-CNN, it simply requires the image or video to travel through its network once.
5. **Firebase:** Is a Backend-as-a-Service a viable option? (BaaS). It offers a number of tools and services to assist developers create high-quality apps, expand their user base, and make money. It is based on the infrastructure of Google.
6. **Firebase Database:** The real-time database maintained by Firebase is a cloud-based database. Data is stored in JSON format and synchronized in real time to all connected clients.
7. **Object Detection:** Is a computer vision approach for identifying and locating objects in images and videos. Object detection can be used to count objects in a scene, determine and track their precise locations, and precisely label them using this type of identification and localization.

**1.7 Organization of the Dissertation/Thesis**

This thesis is organized as follows: chapter one covers the study's background, the research topic, the study's objectives, the study's scope and constraints, and the definition of essential terminology. The second chapter describes existing approaches and provides a review of important works pertinent to this thesis. The third chapter focuses on a comprehensive approach to design and research technique. This chapter also goes through the system architecture, requirements, and specifications in detail. The results and further discussion are presented in Chapter 4. The conclusion and summary of the full thesis report are presented in Chapter 5.It also gives the engineering implication of the findings, contribution to knowledge, recommendations as well as suggestions for further studies.

**CHAPTER TWO**

**REVIEW OF RELATED LITERATURE**

**2.1 Introduction**

In video analysis of the number plate image, computer vision, character recognition, and license plate recognition algorithms all play a role. As a result, they serve as the foundation for any Automatic Number Plate Recognition (ANPR) system. A camera, a frame grabber, a computer, and custom created software for image processing, analysis, and recognition make up the system for autonomous car license plate recognition.

**2.2 Internet of Things (IoT)**

It is the future technology that will bring the entire planet together in one location. All objects, things, and sensors can be connected to share data collected in multiple locations and to process/analyse that data in order to coordinate applications such as traffic signalling, mobile health monitoring in medical applications, and industrial safety assurance methods, among others. According to analysts, 50 billion objects will be connected to the Internet of Things by 2020. IoT provides a diverse set of device connectivity options, as well as a variety of protocols and application features, to enable complete machine-to-machine interaction.

Because of the Internet of Things, existing technologies such as home automation, wireless sensor networks, and control systems will become more efficient and smarter. The Internet of Things has a wide range of applications. Medical applications, for example, monitor a patient's health and transmit data wirelessly. IoT is also used in the current development of wearable instrumentation. Smart wristbands, navigation pills, and other wearable equipment are examples. To update health information or control the gadget using a smart phone, all of these ways require an internet interface. The internet of things is also important in media applications for advertising and information exchange around the world. Manufacturing operations also necessitate IoT for supply chain management and digital control systems for process monitoring. In the case of monitoring applications in space, the requirements of IoT technology and geographical parameters are always significant. When acquiring data from things, the geographical dimensions of the objects are equally significant. The most widely used field of automation is IoT in car applications and traffic maintenance. A vehicle's autonomous gadgets should be connected to the cloud so that the car's health may be updated over time. People can easily identify the shortest path to their destination from traffic monitoring systems and navigate automatically by checking all other directions by linking their automobiles and traffic signalling systems to the internet (Sahu, Gaikwad, Sandage and Shinde, 2017).

**2.3 Existing Methodologies**

The ANPR problem has been solved for many years, and several approaches have been offered to tackle it. The seven algorithms used by ANPR systems are given below:

* 1. General pre-processing, which entails deblurring, scaling, and, in general, converting to greyscale.
  2. Plate Localization, which entails scanning the image and isolating the Representative Pixel (RP).
  3. Plate Reorientation and Resizing accounts for plate skew and resizes the isolated plate to an optimal processing size.
  4. Image Intensity Transformation, which adjusts the image's brightness and contrast.
  5. Noise Removal removes unnecessary information from RP.
  6. Character Segmentation identifies constituents’ symbols in RP image.
  7. Character Recognition matches the segmented characters to priorly saved templates.

There are a wide range of strategies available, from simple algorithms to more complex ones. For each method, issues such as performance, execution time, and platform are given when available. It is important to note that there is a lack of consistency in the way methods are evaluated, therefore declaring which techniques genuinely display the best performance is not suitable. Matrix labs created an ANPR system based on car license plates. The technology employs image processing techniques to identify the car from the computer's database. The system performs admirably under a wide range of situations and with various types of number plates. A car license plate detecting system was created utilizing the template matching technique. Although the system achieved 95 % accuracy, it could only distinguish white background plates (Tejas *et al.,* 2017).

A system for recognizing a car number plate and a vehicle owner's registration details was also built utilizing the pattern matching technique to achieve high accuracy. The method worked well with any background colour, but mismatched text required edge detection to be identified (Dwivedi, *et al.,* 2016). The general OCR technique was used to create an automatic plate recognition system that could recognize many fonts; however, the system's accuracy was poor. ANPR was developed utilizing image processing techniques (neural network) on smart phones and achieved a 93 percent accuracy rate, but the system was unable to recognize filthy or damaged plate numbers (Wang, 2010).

* 1. **Traditional License Plate Recognition Algorithms**

Traditional license plate identification systems extract features from license plates using image processing, such as edge detection and horizontal and vertical projection. These features are mostly human-visible features that are mathematically modelled by algorithms to provide features that humans seek. Finally, feature comparison is used to classify characters. The following are the four methods for character classification:

1. **Template Matching:** The template that matches the distance between the standard templates is calculated by Wang *et al.,* (2010); Yu and Kim (2000). Only some shooting conditions, such as fixed shooting angles and light sources, are compatible with template matching. Although template matching is quick, it does not take into account changes in the shooting environment.
2. **Machine Learning Classification:** Machine learning approaches such as Support Vector Machine (Wen, Lu, Yan, Zhou, Deneen, and Shi, 2011) and K Nearest Neighbour (Wen, Lu, Yan, Zhou, Deneen, and Shi, 2011) can be used to classify characters directly (Karasu, Altan, Sarac and Hacioglu, 2017). To cope with categorization, this method employs linear equations. Although the method is simple, it is difficult to categorize nonlinear and multi-categories, and the quantity of calculation required is significant, thus it has been gradually superseded.
3. **Neural Networks:** The neural network Anagnostopoulos (2006); (Nukano *et al.,* 2004) is trained to extract characteristics in the same way that traditional approaches do, so that the machine may learn these features in the same way that people do, and then learns to utilize them as a classification foundation to identify new data.
4. **Convolutional Neural Networks (CNNs):** Although it appears simple to humans, the classic license plate recognition method, which takes information from people to train the computer, is difficult to learn and classify for the machine, and the effect is not better than the machine itself. As a result, CNNs are presented as a method for teaching a machine to extract features and classify objects on its own (Li and Li, 2015). Despite the fact that CNN can extract features on its own, it can only classify images and cannot do object localisation. It can only be used at the end of a typical license plate recognition system to recognize a single character on a license plate one at a time (Lin, Lin and Liu, 2018).
5. **Histogram of Oriented Gradients (HOG):** The Histogram of Oriented Gradients is a feature descriptor used in image processing and other computer vision techniques to recognize objects. Gradient orientation of localized sections of an image, such as the detection window and the Region of Interest (ROI), are included in the HOG descriptor technique. One advantage of HOG-like features is their simplicity, which makes the information they contain easier to comprehend.

**2.5 Object Detection-Based License Plate Recognition Algorithms**

The machine learning architecture of object detection is used by an object detection-based license plate identification system (Lin *et al.,* 2019; Abdullah *et al.,* 2018) to find and recognize the characters on license plates. Although the license plate identification system based on object detection has substantially improved the anti-noise ability and large-angle recognition capacity when compared to the classic license plate recognition system, the ability to segment overlapping letters is still insufficient.

1. **You Only Look Once (YOLO):** The term 'You Only Look Once' is abbreviated as YOLO. This is an algorithm for detecting and recognizing different items in a photograph (in real-time). Object detection in YOLO is done as a regression problem, and the identified photos' class probabilities are provided. Convolutional neural networks are used in the YOLO method to detect objects in real time. To detect objects, the approach just takes a single forward propagation through a neural network, as the name suggests. This indicates that a single algorithm run is used to forecast the entire image. The CNN is used to forecast multiple bounding boxes and class probabilities at the same time. There are several variations of the YOLO algorithm. Some of the common ones include tiny YOLO and YOLOv3 **(**Kumthekar *et al.,* 2018).
2. **Mask Region-Based Convolutional Nural Network (R-CNN):** Mask R-CNN is a hybrid of Residual Network (ResNet) (He, Zhang, Ren, and Sun, 2016) and Feature Pyramid Network (FPN) (He, Zhang, Ren, and Sun, 2016). (Lin *et al.,* 2017). It can alternatively be thought of as a hybrid of Faster R-CNN (Ren, He, Girshick, and Sun, 2017) and Fully Convolutional Network (FCN) (Long *et al.,* 2015), with FCN generating a portion of the mask.
3. **ResNet:** ResNet's key benefit is that it uses residual learning to tackle the degradation problem of deeper neural networks. ResNet is a multi-layered architecture made up mostly of four residual blocks that are continually matched to other layers. The residual block will construct a shortcut in the input of each of the two convolutional layers and execute an element-wise addition to the outputs to preserve the influence of the original input, thus solving the problem of deep plain networks being unable to converge. (He *et al.,* 2016).
4. **Faster R-CNN:** Faster R-CNN is a two-stage system with two stages: proposal and detection. ResNet serves as the backbone of Faster R-CNN, and it is used with the Region Proposal Network (RPN) to identify the anticipated bounding boxes. Faster R-CNN produces a feature map using ResNet and delivers it to RPN during the proposal stage. RPN is a full-convolutional network that determines region proposals with varying anchor box sizes using a sliding window. The region suggestions are reset to the feature map of the original ResNet output at the detection stage, after which RoIPool is performed, the resultant RoI is corrected, the final bounding box is obtained, and the classes and boxes are output (Girshick, 2015).
5. **Feature Pyramid Network (FPN):** For the convolution and pooling layer results, FPN generates multiple scales of feature maps. The bottom feature map will generate the RPN prediction output directly, then execute up sampling and element-wise addition to the prior feature map, which is four times larger, before generating the RPN prediction output again. The same procedure is repeated by adding elements one by one to the prior feature map, which is now four times larger, and then predicting the RPN result (Lin *et al.,* 2017)
6. **Fully Convolutional Networks (FCN):** The Completely Convolutional Network (FCN) is a fully convolutional network. FCN is not connected to a fully connected layer like classic CNN. FCN directly labels and classifies each pixel in the image, giving it the potential to perform semantic segmentation (Long, Shelhamer and Darrell, 2015).
7. **Region of Interest Align (RoIAlign):** Faster R-CNN is optimized using RoIAlign. Pooled and quantization on RoI are performed faster using RCNN. Because the RoI Region does not always fall on the intersection of grids, it will result in positional offsets. This problem may not appear to be critical at first glance, but it has had a significant impact on the accuracy of identifying small objects (Lin *et al.,* 2017).
8. **Single Shot Detector (SSD):** SSD is a deep neural network-based approach for recognizing objects in pictures. Over a range of aspect ratios, the SSD technique discretizes the output space of bounding boxes into a collection of default boxes. The approach scales per feature map position after discretization. To naturally manage objects of varied sizes, the SSD network combines predictions from numerous feature maps with different resolutions (Ren *et al.,* 2017).
9. **Spatial Pyramid Pooling (SPP-net):** The SPP-net is a network structure capable of generating a fixed-length representation independent of image size or scale. Pyramid pooling is considered to be resistant to deformations of objects, and SPP-net outperforms all CNN-based image classification algorithms. Researchers can use SPP-net to construct fixed-length representations for training detectors by computing feature maps from the entire image once and then pooling features in random sections (sub-images). This approach eliminates having to compute the convolutional features multiple times (He *et al.,* 2016).

**2.6 Types of Automated Vehicle Verification System (AVVS)**

There are various types of AVVS which include the following:

1. **Optical Character Recognition (OCR):** Both academic research and industry are paying more attention to OCR. It has long been man's ambition to create machines that can perform human duties. Reading documents including various types of text is one such reproduction of human functions. Through the development of sophisticated and strong optical character recognition over the previous few decades, machine reading has progressed from a pipe dream to a reality (Dholakia, 2015). OCR technology allows us to convert various forms of documents into editable and searchable data, such as scanned paper documents, PDF files, or digital camera photographs. OCR systems have become one of the most successful implementations of artificial intelligence and pattern recognition technology. Despite the fact that there are numerous commercial OCR systems available for a range of applications, the existing machines are still unable to match with human reading capabilities at the needed accuracy levels. OCR is a type of machine recognition technique that allows for automatic identification.  Automatic identification is a process in which a recognition system automatically recognizes items, obtains data about them, and puts that data directly into computer systems, without the need for human intervention. External data is collected via analysing photos, sounds, or videos. A transducer is used to capture data, which converts the actual image or sound into a digital file. The file is then saved and can be analysed by the computer at a later time. The keyboard is the typical method of entering data into a computer. Nevertheless, this is not always the best or the most efficient way. The automatic identification may serve as an alternative in many cases. There exist various techniques for automatic identification which cover the needs for different application areas. Some notable technologies and their applications worth mentioning apart from OCR are speech recognition, radio frequency, vision systems, magnetic stripe, bar code, magnetic ink and optical mark reading. These technologies have been actively used in past decades (Sagar, 2017).
2. **Automated Vehicle Tracking and Service Provision System:** An automated vehicle tracking and service provision system including a central controller, a local controller located in each vehicle, the central controller and the local controllers including wireless communication interface for communication of information between the central controller and the vehicle based on fuzzy logic algorithms decision making software. In a preferred embodiment, the local controller includes a processor, a global positioning systems (GPS) sensor coupled to the processor for providing vehicle location in terms of latitude and longitude, a memory coupled to the processor, a plurality of sensors coupled to the processor and adapted to provide information on a plurality of parameters related to the vehicle such as fuel level, collision status, brakes and such like, a user interface coupled to the processor for providing user input from input devices such as a credit card reader, smart card reader or keyboard, a wireless transceiver is coupled to the processor for communicating data from the processor to the central controller and for receiving data from the central controller, and a display (He *et al.,* 2016). Provides for a voice or audio input/output interface coupled to the user interface for providing voice activation of the processor or voice transmission via the wireless transceiver to the central controller.
3. **Automatic License Plate Recognition (ALPR):** In real-world applications, the ALPR system is quite useful. ALPR extracts a vehicle's plate number from a digital camera image. The software model is the most significant component of this system. The software model use series of image processing techniques which are implemented in MATLAB. LPR usually is used for detection of some special vehicles. This feature is mostly focus on the vehicles listed in the blacklist, such as the vehicle fled after the accident, or reported as stolen. As long as the relevant information of license plate is input into dataset, LPR will automatically lead to find out the vehicle information. Once the vehicle information found, the systems will feedback it to police station. The techniques of the license plate recognition have been developed for many years. With the hardware and software improved, the accuracy of license plate recognition has been improved to some extent. But the current accuracy still cannot satisfy all the requirements of the traffic department concerned. The license plate extraction is not a simple process, as under some situations, it is easy for people to judge but it will be very hard for computer to do it (Gupta, 2016).

**2.7 Neural Networks**

Neural Networks are artificial networks used in Machine Learning that work in a similar fashion to the human nervous system. Many things are connected in various ways for a neural network to mimic and work like the human brain. Neural networks are basically used in computational models. Recurrent neural networks, convolutional neural networks, artificial neural networks, and feedforward neural networks are all types of neural networks, and each offers advantages for specific use cases. They all work in a similar way by putting data into the model and allowing it to determine whether or not it has made the correct interpretation or conclusion about a certain data element (Laroca *et al.,* 2018).

Because neural networks are based on trial and error, they require a large amount of data to train on. It's no coincidence that neural networks only became popular after most businesses adopted big data analytics and amassed massive data sets. Because the model's initial iterations contain educated estimates about the contents of an image or sections of speech, the data used during the training stage must be labelled so that the model can see if its assumption was correct. This means that, while many big data companies have a lot of data, unstructured data is less useful. Unstructured data can only be analysed by a deep learning model after it has been trained and reached a satisfactory degree of accuracy, but deep learning models cannot train on unstructured data (Ren, 2017).

**2.8 Deep Learning**

Deep learning is a machine learning and Artificial Intelligence (AI) technique that mimics how humans acquire knowledge. Data science, which covers statistics and predictive modelling, incorporates deep learning as a key component. Deep learning is highly useful for data scientists who are responsible with gathering, analysing, and interpreting massive amounts of data; it speeds up and simplifies the process. Deep learning can be regarded of as a means to automate predictive analytics at its most basic level. Deep learning algorithms are built in a hierarchy of increasing complexity and abstraction, unlike typical machine learning algorithms, which are linear.

Deep learning is an artificial intelligence subset of machine learning that uses neural networks to learn unsupervised from unstructured or unlabeled data. Deep neural learning or deep neural network are other terms for the same thing. Deep learning has progressed in lockstep with the digital era, which has resulted in an avalanche of data in all formats and from all corners of the globe. Big data is gathered from a variety of sources, including social media, internet search engines, e-commerce platforms, and online theaters. This massive volume of data is easily accessible and can be shared via fintech tools such as cloud computing. Deep learning, a subtype of machine learning, employs artificial neural networks at a hierarchical level to carry out the machine learning process. Artificial neural networks are constructed in the same way as the human brain, with neuron nodes connected in a web-like pattern. While typical programs develop analysis using data in a linear manner, deep learning systems' hierarchical function allows machines to process data in a nonlinear manner (Laroca *et al.,* 2018).

**2.9 Materials Used**

The materials needed for this project are divided into two parts which are:

1. Hardware Materials
2. Software Materials

**2.9.1 Hardware Materials**

Hardware materials refer to the physical component of the computer. For this system design, the hardware requirements are as follows:

1. **Raspberry Pi Processor:** The Raspberry Pi is a credit-card-sized single board computer developed in the United Kingdom by the Raspberry Pi Foundation. It was utilized in the proposed ALPR system. The Raspberry Pi is powered by a Broadcom BCM2835 system on a chip (SoC), which comprises an ARM1176JZF-S 700 MHz processor, Video Core IV GPU, and 256 MB of RAM, which was later increased to 512 MB.
2. **USB Camera:** A webcam, sometimes known as a USB camera, is a video camera that transmits its image to a computer or computer network in real time. Unlike an IP camera, which connects to the internet via ethernet or Wi-Fi, a USB camera connects to the computer by a USB cable, FireWire cable, or other similar cable. The webcam got its name from its widespread use as a video camera for the World Wide Web. Security monitoring, computer vision, video broadcasting, and filming social videos are all prominent applications. Webcams are the most affordable kind of video communication due to their low manufacturing costs and adaptability. Because certain built-in webcams may be remotely activated via spyware, they've also become a source of security and privacy concerns.
3. **2.9.2 Software Materials**

For this system design to function effectively and efficiently, the software requirements are:

1. **Python Integrated Development Environment (IDE):** Is a software development program based on the Python programming language.
2. **Firebase Backend Platform:** Google's mobile development platform, Firebase, allows you to easily build and scale your app. It's designed to make it simple to add Google Cloud products as your infrastructure requirements change.
3. **Google Collaboratory (Colab):** Is a Google research product. Colab is a web-based Python editor that allows anyone to write and run arbitrary Python code. It's notably useful for machine learning, data analysis, and education. Colab is a hosted Jupyter notebook service that doesn't require any setup and offers free access to computational resources, including GPUs.
4. **TensorFlow:** Is an open-source machine learning platform that runs from start to finish. It has a large, flexible ecosystem of tools, libraries, and community resources that allow researchers to advance the state-of-the-art in machine learning and developers to quickly build and deploy ML applications.
5. **Keras:** Is a free open-source python framework for constructing and analyzing deep learning models that is both powerful and simple to use.
6. **Labeling:** Is a graphical image annotation tool that allows you to label the bounding boxes of objects in images.
7. **OpenCV:** OpenCV is a large open-source library for computer vision, machine learning, and image processing, and it currently plays a critical part in real-time operations, which are critical in today's systems. It may be used to detect items, faces, and even human handwriting in photos and movies.

**2.10 Review of Related Works**

**2.10.1 Accurate Detection and Recognition of Dirty Vehicle Plate Numbers for High-Speed**

**Applications. Institute of Electrical and Electronics Engineers Transactions**

**on Intelligent Transportation Systems.**

Panahi (2016) presented an online approach for Automatic Number Plate Recognition that is both accurate and fast. Plate detection is the first step in this procedure, followed by character segmentation, and lastly character identification. For the license plate localization process, they employed a tweaked version of the method Random Sample Consensus (RANSAC). A variety of data sets were developed. For training and testing, the Crossroad and Highway  datasets were employed. The crossroad data collection was compiled from a variety of locations. Highway data was gathered from highways and streets. Plate segmentation was completed, followed by the input of a greyscale image and the application of a global/adaptive threshold separated M\*N blocks. The characters are processed after locating the plate region, and the output line is supplied to the RANSAC's character recognition section. Then, to determine whether a component was a character or not, a two-class SVM was utilized. For the three steps, the framework achieves accuracies of 98.7 percent, 99.2 percent, and 97.6 percent, respectively.

**Research Gap:** The system was not language-dependent, and it had an overall accuracy of 97 percent.

**2.10. Dynamic Traffic Management System Using Infrared (IR) and Internet of Things**

**(IoT).  Third International Conference on Science Technology Engineering and**

**Management.**

Rani, Kumar, Naresh and Vignesh, (2017) The traffic management system was implemented , which uses IoT, infrared sensors, and image processing algorithms to regulate traffic dynamically. The data gathered by the IR sensor is subsequently communicated through Wi-Fi to the raspberry controller in the system, which receives it. The Wi-Fi transmitter transmits the data collected by the IR sensor, which is received by the raspberry-pi controller. Based on this data, the red signal's time is dynamically shifted, and the user is informed of the signal's condition as he travels. The Raspberry Pi controller acts as a central console, determining whether the road signal should open or close on one side or the other. The central console collects data from sensors and saves it in the cloud, informing a mobile device of traffic conditions. These signals which have the ‘congestion’ mark will indicate the Raspberry-Pi processor, which was installed inside the signal. The Raspberry-Pi in- structs the traffic controller to show the appropriate signals based on the denseness of the traffic. The data signal given to the traffic light will give ‘green light’ to the congested side (the timer in the congested side is given more seconds than the other).

**Research Gap:** The existing approach has a key flaw in that it rotates the traffic controller clockwise and does not account for traffic density. The traffic density is calculated, and the timer display shifts dynamically. This significant benefit eliminates the possibility of an "unwanted delay" for automobiles in congested areas.

**2.10.3 Localization and Recognition of a Myanmar License Plate Based on Partially Cut**

**Character Structure. International Conference on Information and Communication**

**Technology and Knowledge Engineering.**

The proposed technique (Htay and Gopalakrishnan, 2016) has a pseudo-code that is implemented in Matlab. Colour segmentation is used to transform a colour image to a binary image, large region filtering is used to identify candidate regions regardless of the size of the white region in the binary image, license plate localization, character segmentation, and character recognition are all included. The license plates' Euler number and aspect ratio were used to determine their location. In the absence of skewness correction, partial cut character structure was used to implement recognition, and the system produced good results, even for skewed plates. More than 100 genuine license plate photos were used to test the suggested technique. According to the results, there was a 1percent localization error due to camera resolution and an 8 percent character recognition error due to skewed and deformed plates. In conclusion, the system was one of the best, as it properly recognized 92.7 percent of license plate characters and was able to locate 99 percent of plates.

**Research Gap:** The failure to locate the plate was due to low image quality, and the image enhancing technique was not included in this study.

**2.10.4 Smart Vehicle Number Plate Detection for Different System for Different Countries**

**using an Improved Segmentation Method. Imperial Journal of Interdisciplinary**

**Research.**

Balaji and Rajesh (2017) first chose the best frame and retrieved the license plate, then isolated the characters and identified them individually. The methods employed was simple but effective. First, all of the characters in the image were fragmented (Licence Plate). Finally, each character's identity is established. Each character on the vehicle license plate is recognized using the pattern matching method. To improve number plate detection, they applied an ant colony optimization technique for segmentation. They removed noise, adjusted brightness and contrast, and then transformed the raw image to grayscale to create the binarized image. The Hough transform was used to locate the specific plate region using Hough lines. Finding edges in the image and segmenting each character in the license plate region were also implemented. To eliminate shadow, dirt, and screws, two morphological techniques were used: area opening and erosion. In order to get good performance in number plate detection, the Ant's colony optimization (ACO) technique was utilized for segment. Finally, the template matching method with correlation was employed to recognize each character on the number plate. The removal of image noise was another crucial stage in their algorithm, which was followed by license plate localisation and character segmentation.

**Research Gap:** Some of the photos were unclear, and comparable characters such as 3 O and D; 5 and S; 8 and B, E; O and 0 could not be identified.

**2.10.5 A Real-Time Mobile Vehicle License Plate Detection and Recognition. Journal of**

**Science and Engineering.**

In an outdoor space, Hung and Hsieh (2010) demonstrate an instant and real-time mobile car license plate recognition system. The technology tries to capture the image of the car in front of it and perform instant vehicle license plate detection and recognition using a nonfixed video camera put in the car. To detect license plates, the writers make use of the barking lights' colour properties. First, in the acquired image, the location of two barking lights was discovered. Then, using the probability distribution of the license plate between the two lights, the license plate detecting region was set. This technology may eliminate any environmental interference during license plate detection and recognition, as well as improve the rate of accuracy. Furthermore, the authors applied the Black Top-Hat morphological method to improve the level of separation between the license plate characters. Experiments show that the system can record the vehicle image, detect, and recognize the license plate whether it is daytime, night-time, clear day, rainy day, or in a complex environment.

**Research Gap**: The video camera was mounted in a car instead of a fixed location to catch the license plate of a random vehicle in front.

**2.10.6 Background Subtraction for Static and Moving Camera. Institute of Electrical and**

**Electronics Engineers International Conference on Image Processing.**

Sajid and Cheung, (2015) proposed a multimode background subtraction is a novel detecting method that has been designed using an improved background subtraction algorithm (MBS V0). Originally, this algorithm was created to deal with abrupt lighting changes. The new technique runs numerous pixel-wise models of the scene with frame level constraints, allowing it to work with both static and moving cameras. The new version has been improved with adjustments at various stages of the process, including the selection of the best colour space, clustering of training photos for the Background Model Bank, and parameters for each colour space channel. This has allowed the technique to be used to solve a wide range of change detection problems, including camera jitter, dynamic backgrounds, IOM (Intermittent Object Motion), shadows, bad weather, thermal, and night films. A thorough analysis indicates the algorithm's superiority above the current state of the art. Unlike other techniques that use a single backdrop model, the authors used a frame level constraint to create numerous pixel-wise BG representations of the scene. As a result, the suggested system can now account for inter-pixel spatial relationships, making it applicable to both static and moving camera settings.

**Research Gap:** The model updating mechanism is unavailable in this MBS V0.

**2.10.7 Myanmar Character Extraction from Vehicle Images Using Aspect Ratio and**

**Bounding Box. International Workshop on Advanced Image Technology (IWAIT).**

The license plates were used as the experimental material for a Myanmar character extraction in Khin, Phothisonothai and Choomchuay, (2018). Pre-processing and extraction are the two key processes in the proposed system. The incoming colour photographs were transformed to grayscale photos and binarized by thresholding during pre-processing. These binarized photos were cleaned of noise. These photos were then utilised in subsequent processing processes. Myanmar's letters and digits are extremely similar and complex. As a result, the image processing toolbox function is used to segment and extract the characters from the license plate image during the character extraction step. Character extraction was done with a bounding box in the area, and character extraction was based on character and image sizes. The goal of this phase was to prepare Myanmar automobiles and extract their numbers and letters. Finally, the characters were accurately extracted with a 90 percent accuracy rate, according to the test findings.

**Research Gap:** The proposed model had 10 percent extraction failure.

**2.10.8 Application-Oriented License Plate Recognition. Institute of Electrical and Electronics**

**Engineers Transactions on Vehicular Technology.**

For character segmentation, the Maximally Stable Extremal Regions (MSER) was proposed by Hsu, Chen and Chung, (2013). The local binary pattern function for character identification was extracted and classified using a linear discriminant analysis classifier. A method for directly detecting the complete license plate without segmentation has been proposed thanks to the development of deep neural networks. To better manage application-oriented LPR, a three-module approach for plate detection, character segmentation, and identification is provided, with settings that may be adjusted for different applications. The application-oriented license plate (AOLP) database was created and made available to the research community in order to evaluate the performance of the suggested solution. Experiments show that the suggested approach outperforms many earlier solutions, and that solutions with settings tailored for different applications are better at solving LPR.

**Research Gap:** Character segmentation, on the other hand, is a challenging operation that can be hampered by uneven illumination, shadows, and image noise. It has a direct bearing on plate recognition. Even if a sophisticated recognizer is used, the plate cannot be successfully identified if the segmentation is incorrect.

**2.10.9 PixTextGAN: Structure Aware Text Image Synthesis for License Plate Recognition.**

**Institution of Engineering and Technology Image Processing.**

To generate synthetic license plate images with adequate text quality,Wu, Zhai and Cao, (2019) presented a unique PixTextGAN that uses a customizable architecture that generates specific character shapes for different text regions. To generate synthetic license plate images with adequate text features, the authors propose a new PixTextGAN that uses a customizable architecture that generates specific character structures for different text locations. Rather than collecting a large amount of annotated data, license plate images were created using generative adversarial networks (GAN). In particular, a structure-aware loss function is described in order to preserve the key characteristics of each character region and so accomplish appearance adaptation for better recognition. In text image synthetisation, qualitative and quantitative trials show that the authors' suggested technique outperforms state-of-the-art GANs. Further license plate identification experiments on the ReId and CCPD datasets show that adding PixTextGAN's synthesised images can considerably enhance recognition accuracy.

**Research Gap:** This system needs more data to achieve good results.

**2.10.10 Real-Time License Plate Detection and Recognition Using Deep Convolutional Neural**

**Networks. Journal of Visual Communication and Image Representation**.

Silva and Jung, (2020) proposes an end-to-end ALPR technique based on a hierarchical Convolutional Neural Network (CNN). The suggested method's main idea is to use two passes on the same CNN to identify the vehicle and the license plate region, and then use a second CNN to recognize the characters. The recognition CNN investigates the use of synthetic and enhanced data to deal with restricted training datasets, and our findings suggest that the augmentation procedure improves the recognition rate dramatically. The authors also introduced a novel temporal coherence approach for better video OCR output stabilization. This method was evaluated on publicly available datasets of Brazilian and European license plates, and it outperformed both competitive academic methods and a commercial system in terms of accuracy.

**Research Gap:** The architecture utilized in this study was too deep, too wide, and had too many parameters; it had around 8 convolutional layers and 3 million parameters. As a result, such architecture could not be used on the edge since it would take too long for a single frame to be processed on a Raspberry Pi or other small computer. It was necessary to use an Nvidia GPU for it to work properly.

**2.10.11 Vehicle License Plate Recognition Based on Extreme Regions and Restricted**

**Boltzmann Machines. Institute of Electrical and Electronics Engineers Transactions on**

**Intelligent Transportation Systems.**

Gou, Wang, Yao and Li, (2016) developed an approach for recognizing automobile license plates that uses character-specific Extremal Regions (ERs) and Hybrid Discriminative Restricted Boltzmann Machines (HDRBMs). Top-hat transformation, vertical edge detection, morphological operations, and numerous validations are used to conduct coarse License Plate Detection (LPD). Then, in license plate candidates, character-specific ERs are retrieved as character regions. Character segmentation and coarse-to-fine LPD are done simultaneously after a proper selection of ERs. Finally, the characters are recognized using an HDRBM offline trained pattern classifier. The proposed method is unaffected by variations in lighting and weather over the course of a day or 24 hours. The usefulness of the suggested approach in complicated traffic conditions is demonstrated using experimental results from large data sets.

**Research Gap:** The recognition rate is strongly dependent on correct character region extractions in the recognition step. It is possible to move some extracted or inferred character regions with low probabilities. Even when the authors used deep architectures to extract features from raw pixel data, they were unable to get optimal results.

**2.10.12 License Plate Recognition System. International Conference on Advance Computing**

**and Innovative Technologies in Engineering**.

Ali, Rathor, and Akram, (2021) presented a fusion technique for improving the quality of vehicle photos, after which the license plates were retrieved and the characters present on the number plate were isolated and identified using an artificial neural network. The suggested approach comprises of two steps: plate number identification and character identification. In the plate number detection method, the number plate is identified from a photograph, and the segmented plate is then transmitted to the plate identification process in the second phase to determine the characters and numbers. The capacity of a neural network to accurately recognize characters on a license plate has a chance of 95 percent even in the presence of noise with a 50 percent density, according to this experiment.

**Research Gap:** During inclement weather, license plate recognition systems are ineffective. As a result, all security mechanisms will be disabled, and manual surveillance will be required.

**2.10.13 Plate Character Recognition Based on Gaussian-Hermite Moments. Second**

**International Workshop on Education Technology and Computer Science.**

Ma, Pan, Wang, (2010) developed a new approach for recognizing license plate characters by employing 2D Gaussian-Hermite Moments (GHMs) of various orders as the input vector of a back propagation neural network with 231 GHMs features. Before recognition, our approach captures the plate picture using pre-processing methods. The smoothing kernel was the gaussian function to make it less sensitive to noise and to avoid the artifacts introduced by the window function's discontinuity. Orthogonal moments with a smoothing window function were used here, and the smoothing kernel was the gaussian function to make it less sensitive to noise and to avoid the artifacts introduced by the window function's discontinuity. The system functioned with a variety of lighting, plate sizes, and dynamic backgrounds. This analysis reveals that orthogonal moment's base functions of different orders have different numbers of zero crossings and very different shapes, allowing them to better reflect image features based on different modes, which is useful for pattern analysis, shape classification, and moving object detection. This technology has the potential to be used in video retrieval and other video information processing domains.

**Research Gap:** The project is still in its early stages of development.

**2.10.14 License Plate Recognition System. International Conference on Fuzzy Systems and**

**Knowledge Discovery.**

Wand and Liu, (2015) The neural network concept was used in a license plate recognition system, according to the proposal and the character of a license plate is used to identify it. The proposed system has established a corresponding workflow and completed it through the PC in order to make the system fast and precisely. The photographic lens will return sequential still photos when employing license plate recognition systems. Before image processing, the system involves numerous aspects, including a core algorithm, detection and capture license plate location, license plate cutting element, license plate element feature extraction, license plate identification, and other components. It count the number of colour changes in the image of each horizontal line while staring at the license plate. The license plate positioning can be discovered by using the qualifying threshold. The initial size of the localization results region will be used to appraise abnormal number plate detection, conditional iteration binarization, changed, and research. Whereby the plate not only achieves better locations, but also improves the cutting plate element's accuracy. Authors employed comparison properties of the target element and backdrop to detect colour change, identify element borders, and cut plate elements in the plate element cutting section. The feature table's experimental results have a 9 percent ability to identify, and test data can be discriminated independently over approximately 90 percent of the license plate components. The findings of the experiment show that the image has an excellent recognition rate.

**Research Gap:** The feature table's experimental results have around 9 percent of the ability to identify, and test data can be discriminated independently over about 90 percent of the license plate components.

**2.10.15 A Lightweight, High-Performance Multi-Angle License Plate Recognition Model.**

**Proceedings of the 2019 International Conference on Advanced Mechatronic Systems.**

A lightweight and high-performance multi-angle license plate character recognition model was proposed by Lin and Wu, (2019), which minimizes the computational complexity of standard license plate recognition. As training data, the authors collect a huge number of license plate photos from various locations, angles, and sizes. The proposed model can recognize the license plate with a tilt of 60 degrees and has an overall recall rate of 84.5 percent, according to the experimental data. This model's computation time is lowered by 61percent when compared to Tiny-YOLOv2, with only a little recall cost.

**Research Gap:** The overall recall rate was 84.5 percent.

**2.10.16 Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation.**

**Institute of Electrical and Electronics Engineers Conference on Computer Vision and**

**Pattern Recognition at Columbus.**

Girshick, Donahue, Darrell and Malik, (2014) suggested a simple and scalable detection technique that enhances mean average precision (mAP) by over 30 percent compared to the previous best result on VOC 2012, with a mAP of 53.3 percent. The authors used a method of sized sliding windows, which results in a lot of wasted processing. This method combines two key ideas: (1) high-capacity convolutional neural networks (CNNs) can be used to localize and segment objects from bottom-up region proposals, and (2) when labelled training data is scarce, supervised pre-training for an auxiliary task followed by domain-specific fine-tuning yields a significant performance boost. Due to the fact that region suggestions was integrated with CNNs. R-CNN: Regions with CNN Features was the name of the method. To save time and effort, the regional CNN (RCNN) predicts roughly 2,000 to 3,000 regional proposals using selective search, then uses CNN models to extract characteristics from regional proposals, and finally uses SVM to complete classification. Following the classification, RCNN uses the bounding-box 5regression to improve the detection outcomes.

**Research Gap:** The RCNN has two significant flaws. The first is that each area proposal in the RCNN is required to pass the CNN along, resulting in a huge number of repeating computations for each image. The second issue is that three different models must be trained individually. A CNN for generating image features, a classifier for class prediction, and a regression model for refining the bounding boxes. This makes training RCNN incredibly tough.

**2.10.17 An Efficient License Plate Recognition System Using Convolution Neural Networks.**

**Proceedings of Institute of Electrical and Electronics Engineers International**

**Conference on Applied System Innovation.**

To eliminate false positives on plate detection, a hierarchical license plate recognition system was devised by Lin, Lin and Liu, (2018) which first identifies cars and then retrieves license plates from those vehicles. The character identification of hazy and opaque images is then improved using convolution neural networks. The YOLOv2 model and support vector machines work together to record license plates with great precision. Character recognition accuracy is also good for the LPRCNN model. A CPU host and a GPU device were installed in the test environment. The host is powered by an Intel® CoreTM i7-4790 processor with four cores running at 3.60 GHz and two hardware threads per core. The GPU in the device is an NVIDIA® GeForce® GTX Titan X with 3,840 CUDA cores and 12GB DDR5X DRAM. Ubuntu 16.04 LTS was used as the operating system. The methodology yields a vehicle detection rate of 96.12 percent and a plate detection rate of 94.23 percent, according to test findings. The authors achieved 99.2 percent character recognition accuracy using LPRCNN. This demonstrates that the suggested license plate recognition system outperforms standard license plate detection systems in terms of accuracy and performance.

**Research Gap:** The RCNN requires each region suggestion to forward the CNN, resulting in a huge number of repeating computations for each image.

**2.10.18 Automatic License Plate Recognition Using Python and Opencv. Department of**

**Computer Science and Engineering, MES College of Engineering.**

Using Python and OpenCV, Sajjad (2010) created a system to recognize plate numbers. The idea was executed utilizing a real-time embedded technology that automatically recognizes vehicle number plates. To capture the image, the system used python software and image processing algorithms from the OpenCV package. It was also used to resize the image, execute error checking, and convert the image to grayscale during pre-processing. To maintain image quality, threshold techniques were employed to binarize the image, and character segmentation was accomplished by scanning the image and cropping out the white section by partitioning the image lines into individual characters using image scissoring.The system is simple to use because it makes use of free open-source software tools that are simple to use, as well as computer vision to convert still images into information that computers can understand.

**Research Gap:** The system, on the other hand, was unable to detect moving vehicles and lacked a database to store the data acquired.

**2.10.19 Accurate Detection and Recognition of Dirty Vehicle Plate Numbers for High-Speed**

**Applications. Institute of Electrical and Electronics Engineers Transactions on**

**Intelligent Transportation Systems.**

Panahi and Gholampour (2016) developed a system for real-time applications that recognizes dirty and unclear vehicle number plates. Variations in weather conditions, bad illumination, changing traffic scenarios, and high-speed vehicles all affect vehicle detection, hence the system was designed to match real-time conditions. To detect car plate numbers, the system used a combination of hardware platforms and sophisticated algorithms. Images recorded from crossroads in various settings, including day and night, as well as other conditions, were utilized to create data sets. Plate detection was used to identify the vehicle plate number, which was subsequently divided into individual characters before character recognition was performed. The system's weakness was its inability to detect moving automobiles. The system's advantages were its high accuracy results of 98.7 percent, 99.2 percent, and 97.6 percent, as well as its multiple real-world applications.

**Research Gap:** Plate identification accuracy was predicated on vehicle plates with at least three legible characters.

**2.10.20 Recognition of Vehicle Number Plate Using Raspberry Pi. International Research**

**Journal of Engineering and Technology.**

Kumthekar *et al.,* (2018) developed a Raspberry Pi-based system that detects plate numbers automatically. The system proposes using the Raspberry Pi CPU to construct an automated plate number recognition system that uses Optical Character Recognition (OCR) to decipher information on photographs of a car plate number. For authentication reasons, the Raspberry Pi processor processes and verifies the image taken. After that, the suggested system split the license plate and detected each fragment particle. The system extracts the number plate utilizing an open ALPR design to reach the final result. After that, the output image was manually compared to the input image. The system utilized was unable to detect moving objects and was unable to improve image quality.

**Research Gap:** However, the system was unable to quickly detect the plate number and interpret the image of the number plate due to a research gap.

**CHAPTER THREE**

**MATERIALS AND METHOD**

**3.1 Introduction**

Vehicle check points along Nigerian roads are usually considered inconveniencing by road user and commercial drivers not only for the traffic lock usually caused by them, but also for the delay that comes with it. The system used was incapable of detecting moving objects and of improving image quality. However, due to a research gap, the system was unable to swiftly detect the plate number and understand the image of the number plate. Therefore, there is a need to develop an Automated VehicleVerification system (AVVS) using an OCR that will be able to identify and verify vehicles with a database platform as one of the solutions to this problem.

**3.2 System Description**

Prior to the deployment of the application, a comprehensive database of all vehicles and their properties is created and a list of faulted vehicles is formed from the database. License plate is used as the primary key in the database as it is specific to each vehicle. Cameras are placed at different checkpoints. The camera detects and recognizes the license plate miles away from the checkpoint to enable the policemen flag down the faulted vehicle once it reaches the checkpoint.

Once a vehicle with a flagged license plate passes through any of these strategically positioned cameras, an alert is displayed on the screen of the device at that checkpoint.

**3.3 Functional Decomposition**

The modules that breakdown the system components are produced in stages and assessed until they meet the system's needs and objectives. The system is divided into three modules:

1. Database Management System (DBMS)
2. License Plate Detection (LPD)
3. Optical Character Recognition (OCR)
4. Graphical User Interface (GUI)

**3.3.1 Database Management System (DBMS)**

As illustrated in Figure 3.1, the database used in this system is Firebase. This is because Firebase maintains all data in the database in real time, data interchange to and from the database is simple and quick. A user (policeman) can make changes to the database in real-time thereby allowing every other user to see the changes being made immediately. This implies that the work of each stationed policeman is not only flagging down faulted vehicles but adding and removing vehicles from the flag list. A user-friendly interface is provided for this purpose. Firebase data is stored in JSON, which has a nested structure and can be represented using a tree. The tree representation for two different license plates “ka243fa” and “lm243fa” is shown in Figure 3.1. From the diagram, each parent: “ka243fa” and “lm243fa” have children: “car”, “engine\_no”, “flagged” and “owner” and each child: “car” and “owner” have grand children: (“model” and “year”) and (“address”, “name” and “phone”) respectively.



**Figure 3.1: Screenshot of a firebase structure sample.**

**Source: Compiled by the researcher.**

**3.3.2 License Plate Detection (LPD)**

The preferred convolutional neural network model for this project was small YOLOv3. The reason for choosing tiny YOLOv3 over other CNN models is that it provides various advantages for this system. Tiny YOLOv3 is a real-time object identification model that can recognize and locate license plate photos accurately. For this problem, several image processing approaches have been used; while they worked for some images, they did not scale for the entire dataset and were nowhere near as precise as tiny YOLOv3 for this task. Another advantage of utilizing tiny YOLOv3 is that it generalizes effectively, allowing it to perform exceptionally well on data it has never been trained on. Tiny YOLOv3 can also effectively filter out background noise from the actual data. Tiny YOLOv3 was chosen to be used in this project for these reasons.

Based on the small YOLOv3 model, this network has 106 completely convolutional layers. It activates itself by using the leaky unit. Tiny YOLOv3's loss function is complete intersection over union (CIOU). Finally, the intersection over union (IOU) between the predicted box and the ground truth box is the confidence Ci. Additionally, if any item is discovered, pi(c) represents the class probability for that object.

**3.3.2.1 Dataset Collection and Training**

The dataset for this project had to be acquired manually because there was no existing collection of Nigerian license plates. This had the benefit of having several license plate image orientations for a more effective model. We recorded live recordings of vehicles on the road with 18-megapixel cameras for our dataset. The still images were recovered in colour mode with standard Joint Photographic Experts Group (JPEG) format by sampling the video footage at a rate of one image per second. There were 2,546 photos for training and 360 images for testing in the dataset. The dataset was manually tagged before the training began, using python labelling to create bounding boxes. To avoid overfitting, the dataset was manually supplemented. The image sizes were randomly translated and scaled up to 20 percent for this purpose. In the HSV colour space, the saturation and exposure of the original photos were modified at random by 30 percent. The photos were also given some random noise to make them more durable.

**3.3.2.2 Dataset Training**

There are 106 completely convolutional layers in the model. The model was trained for 3000 epochs, which took roughly 7 hours, to detect the license plate. The model's hyperparameters were similar to the original YOLO model at the time of training, with Batch size = 64, Subdivisions = 2, Momentum = 0.9, and Decay = 0.0005. When validation accuracy did not improve in a few preceding epochs, the Learning rate began at 0.001 and gradually dropped. Google Collaboratory GPU was used for the training.

**3.3.3 Optical Character Recognition (OCR)**

The optical character recognition aspect of this project utilizes one of Google Cloud Platform's (GCP) services for recognizing texts in photos (Google Cloud Vision API). The Google Cloud Vision API makes it simple for developers to integrate vision detection capabilities like picture labeling, face and landmark identification, optical character recognition (OCR), and explicit content tagging into their apps. Following the detection of the license plate, the object in the bounding box is given to the Vision API, which recognizes all of the text in the object, including the state, state motto, plate number, and country.

* 1. **Materials**

The materials needed for this project were divided into hardware materials and software materials.

**3.4.1 Hardware Materials**

Hardware materials refer to the physical component of the computer. For this system design, the hardware requirements are as follows:

1. Laptop
2. Raspberry Pi
3. USB Camera
4. Internet Infrastructure/WLAN

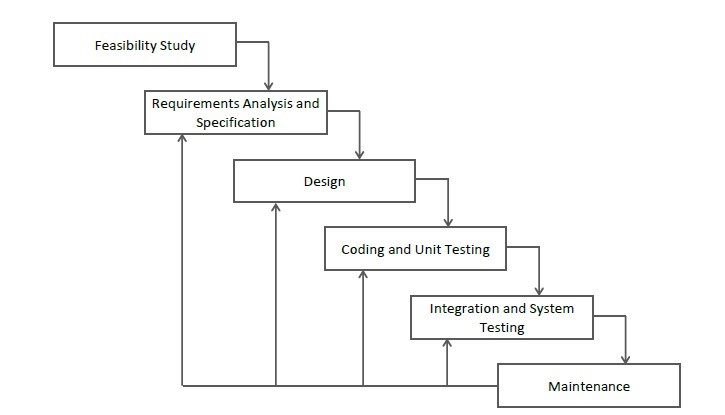
**3.4.2SoftwareMaterials**

For this system design to function effectively and efficiently, the software requirements are:

1. Python Integrated Environment
2. Firebase Backend Platform
3. Google Collaboratory
4. Tensorflow
5. Keras
6. labelling
7. OpenCV

**3.5 Software Process Model**

In the creation of this system, the iterative waterfall model was adopted. The fundamental distinction between the iterative waterfall model and the classical waterfall model is that the iterative waterfall model provides feedback pathways from each step to the phases before it. Figure 3.2 depicts the feedback routes introduced by the iterative waterfall approach.



**Figure 3.2: Iterative waterfall model.**

**Source: WiKi (2020).**

When faults are discovered at a later stage, these feedback pathways allow programmers to remedy errors made earlier in the process. The feedback channels allow the phase in which errors are committed to be modified, and the modifications are reflected in subsequent phases. However, there is no feedback path to the stage – feasibility study, because once a project has been undertaken, it is difficult to abandon it.

**3.5.1 Feasibility Study**

The term "feasibility" refers to the process of determining whether or not a project is worthwhile. The feasibility study is the method used to arrive at this conclusion. For this project, a number of factors such as cost implication and application of the system were put into consideration before drawing the conclusion that the project was feasible.

**3.5.2 Requirement Analysis and Specification**

Requirement analysis and specification identify, analyse and model the functionality or “whats’s” of a prospective software system. Requirement specification is divided into functional and non-functional requirements.

**3.5.3 Functional Requirement**

The basic system behaviour is defined by functional requirements. They are essentially what the system does or does not do in response to inputs, and can be understood of in terms of how the system responds to inputs. The following are some of the project's functional requirements:

1. The system shall contain camera equipment/system for capturing license plates on passing vehicles that are moving or standing still.
2. The proposed solution must be capable of reading correctly 95 % or more of the license plates that are scanned.
3. The quality of license plate recognition must not be affected in different lighting conditions.

**3.5.4 Non-Functional Requirement**

A non-functional requirement is one that sets criteria rather than specific behaviours that can be used to judge the system's operation. Speed, performance, and reliability are some of the project's non-functional requirements.

**3.5.5 Design**

During the first iteration of design phase, the model was built using YOLOv3. This model met the functional requirements but failed to meet some non-functional requirements like speed and performance. Although the performance was good (about 79 percent accurate), it could be better. The speed of the model at this first iteration was reasonably slow and could not hold up on a real-time system. During the second iteration, the algorithm used was changed from YOLOv3 to tiny YOLOv3. This increased the performance of the model considerably to about 93 percent and the speed of the model had improved but still was not good enough for a real-time scenario.

The next iteration focused primarily on the speed of the model. After many trials and errors to improve the speed, I found out that applying parallelism and concurrency gave the best speed output for real-time deployment.

**3.5.6 Coding and Unit Testing**

The system is broadly divided in for four units, each of which was coded and tested individually. The database management system used is firebase. During the first iteration, the data entry was hardcoded into the database for practice. On subsequent iterations, the entry was made to come directly from the GUI. The optical character recognition unit was carried out using Google cloud vision API. The first iteration for this unit was done on Colab which ran smoothly, but since Colab would not be used in the deployment, this led the next iteration which was done locally. The graphical user interface unit was coded and tested locally. This unit was not iterated.

**3.5.7 Integration and System Testing**

At this phase, all four units were integrated into one to form the whole system; from the camera scanning videos for license plate to an alert being displayed for flagged license plates. The Graphical User Interface (GUI) was linked to the database so that entries could be made automatically and the Licensed Plate Detection (LPD) unit was linked to the Optical Character Recognition (OCR) engine on the cloud which was linked back to the database for querying. While the entire system was tested, it was discovered that after the first license plate was being detected and recognized in a couple of seconds, subsequent license plates would only be detected as fast but would take minutes to be recognized. To solve this problem, concurrent and in parallel programming were introduced. This reduced the queuing time for the recognition of subsequent plates thereby increasing the overall speed of the system.

**3.6 Concurrency and Parallelism**

When two jobs are being executed at the same time, this is known as concurrency. It could be the case that a program is working on multiple tasks at the same time. Parallelism, on the other hand, is the art of dividing jobs into subtasks that can be completed at the same time. It differs from concurrency in that it involves the simultaneous occurrence of two or more events.

Concurrency is generally thought to be a broader term than parallelism. To put it another way, it's performing several things at once. In practice, the contrast between the two ideas has a specific angle, notably in Python. Concurrency is sometimes interpreted as "managing" numerous jobs at the same time. In reality, those jobs don't all complete at the same time. They skilfully switch roles. Parallel execution, on the other hand, refers to running numerous jobs at the same time, or in parallel. The parallelism allows numerous cores on a single system to be used.

**3.6.1 Concurrency in Python-Multithreading**

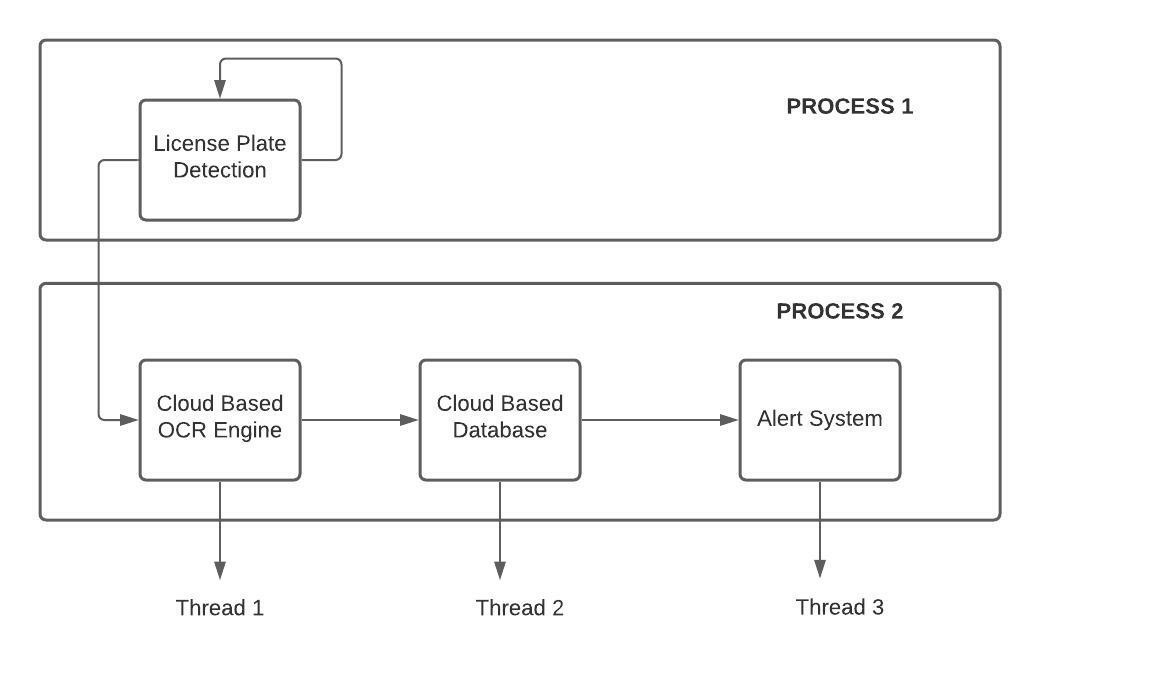
Multithreading refers to a CPU's capacity to manage the use of an operating system by running numerous threads at the same time. Multithreading's fundamental goal is to establish parallelism by splitting a process into several threads. In some languages, concurrency and parallelism may be a matter of semantics (where threads can achieve true parallelism), however, that is not true in Python. In Python, concurrency is the execution of many tasks at the same time with one process. With multithreading in Python, the relevant concept is concurrency.

In one process, there can be many threads (multithreading), and each thread is dealing with a different task. Concurrently, many tasks are being worked on. However, these threads are not working in parallel. A job for a thread is executed a little, and then the context is quickly switched to another thread, and so on until all tasks are completed. Due to this rapid switching, there is an illusion of simultaneity, but it remains only an illusion as threads are not working in parallel, therefore the execution of tasks is not simultaneous. The reason for this behaviour is due to the Global Interpreter Lock (GIL). It is a relic of Python’s past that prevents two threads from executing simultaneously in the same process.

**3.6.2 Parallelism in Python-Multiprocessing**

The employment of two or more CPUs in a single computer system is known as multiprocessing. It is the most effective method for maximizing the performance of our hardware by utilizing the entire number of CPU cores available in our computer system. In contrast with concurrency, parallelism in Python is truly executing tasks simultaneously. This is done by bypassing the GIL. One specificity of the GIL is that it is process-specific. This means that for each process, there is a GIL. Therefore, although only one thread can execute at any point in time within a process, multiple processes can execute tasks in parallel without violating their process-specific GIL. Multiprocessing is done by utilizing multiple cores on the hardware, as opposed to multithreading which uses only one core.

To achieve true parallelism in Python, a task can be split into smaller tasks, each running parallel on different processes on different cores. Figure 3.3 simply shows how concurrency and parallelism is employed in this work. Process 1 and process 2 are ran in parallel while the three threads in process 2 are ran concurrently.



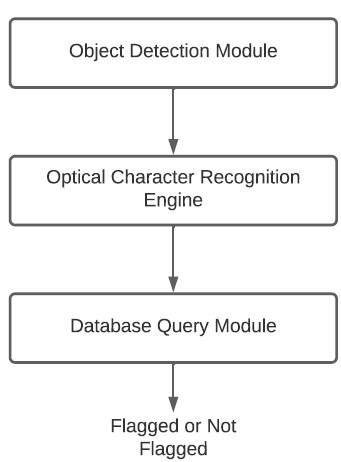
**Figure 3.3: Concurrency and parallelism illustrated.**

**Source: Compiled by the researcher.**

Using a linear straight forward pipeline for the integration of this system proved to not be the most suitable approach. Implementing a linear pipeline means the license plate recognition engine needs to process a frame and extract the plate number ROI for character extraction, during the time which the OCR engine is extracting the characters, the LPR engine is idle, after the OCR engine extracts the characters from the plate number ROI it passes the characters to the database model , the task of the database module is simply to query the database for the detected characters, however during the few seconds it does this querying the OCR engine and LPR engine remain idle, at this point the flaws of such a design are very hard to ignore, the system would be very inefficient, also this linear pipeline significantly increased the processing time, this is due to the fact that my real-time database and OCR engine were dependent on internet connection and the network speed, also there was a lot of idle time during which the various modules of this pipeline were idle while there was data to be processed.

Multiprocessing to split the data processing pipeline into two different processes, the first process which uses the first core of our computer to extract a plate number ROI and drawing a bounding box over the ROI while displaying each frame, the second process is responsible for all network related task , the OCR engine and database query module are on this process, data is shared between the two process with a queue, the LPR process puts ROIs into a queue, while the OCR engine gets the ROI from the top of the queue for processing.

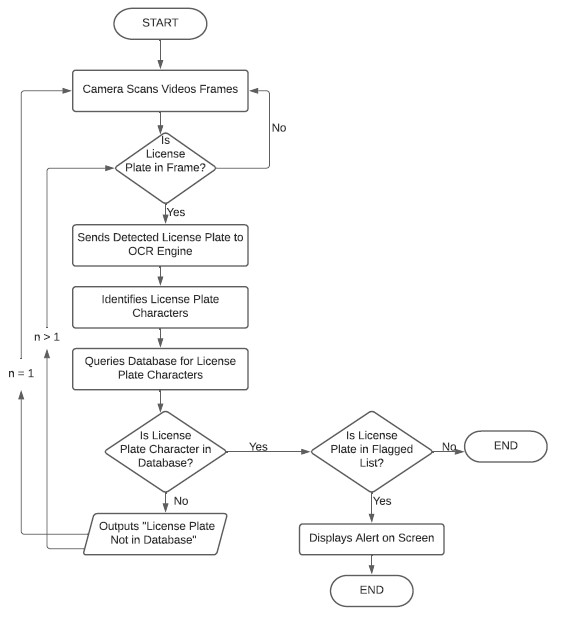
The parallel design helped a great deal in making the system faster, however there are still a lot of ways to exploit concurrency and parallelism to improve the model efficiency. Multithreading techniques was employed in the second process due to the fact that the second process had two tasks of sending request to a server and waiting for a response. Instead of the OCR engine waiting for a response from the server, the database module can be querying the database with the already processed ROI as shown in Figure 3.4.



**Figure 3.4: Modular design.**

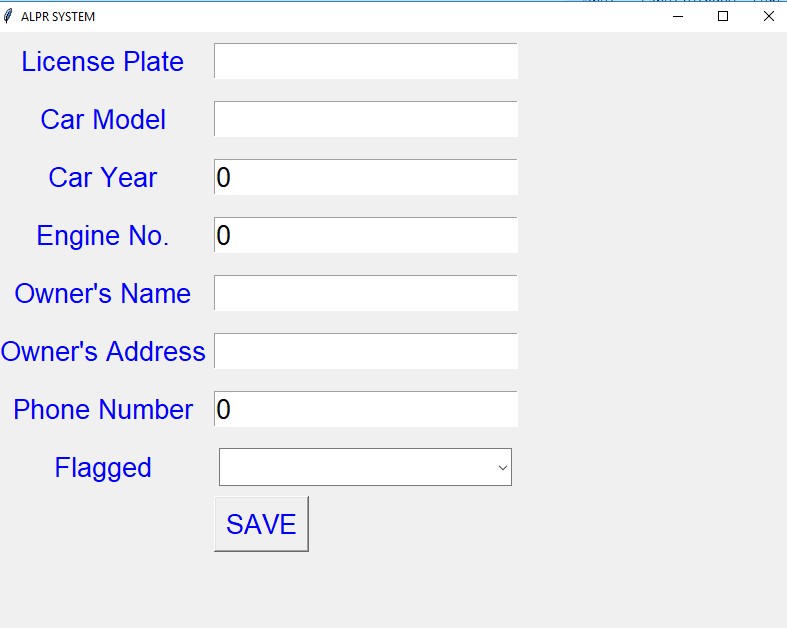
**Source: Compiled by the researcher.**

The flowchart depicting the overall process of capturing Licensed plate using Object Detection Module and Optical Character Recognition is shown in Figure 3.5 and the Graphical User Interface for entering the data of the Licensed Plate is shown in Figure 3.6.



**Figure 3.5: Flowchart.**

**Source: Compiled by the researcher.**



**Figure 3.6: Screenshot of the input graphical user interface.**

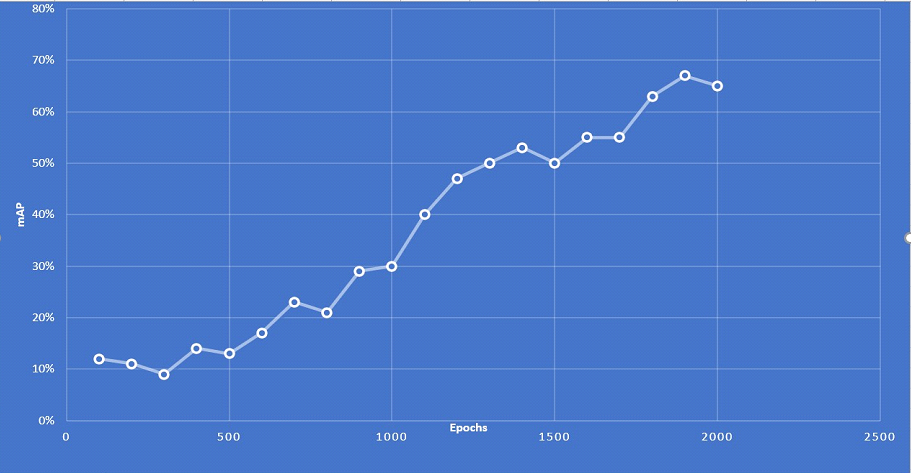
**Source: Compiled by the researcher.**

**CHAPTER FOUR**

**RESULTS AND DISCUSSIONS**

**4.1 Results**

Overall, the system was very effective at carrying out its objectives. Figure 4.1 shows the mean average precision during the training.



**Figure 4.1: Mean average precision during training.**

**Source: Compiled by the researcher.**

**4.2 Effective Range of LPR Model**

During range testing we came to discover that the system had an effective range of 1 to 16 metres after 16 metres the model began to have serious localisation errors, after 20 metres the model did not recognize the plate numbers at all, camera quality had very little influence on plate number localization, however it had great effect on OCR results.

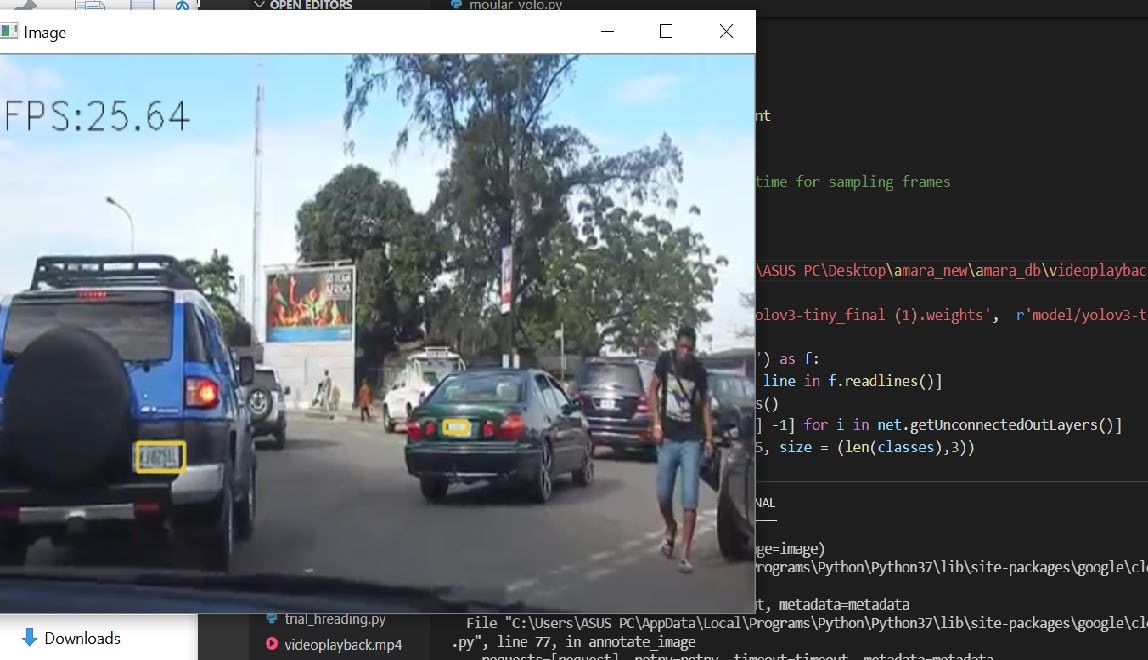


**Figure 4.2: Effective range of LPR model.**

**Source: Compiled by the researcher.**

**4.3 Efficiency and Effectiveness of Parallelism and Concurrency**

The parallel design pushed shown in Figure 4.3, improved the fps from 0.1 frames per second to up to 30 frames per second and an average of 20 frames per second on an eighth generation core i3 running on CPU only, during testing it was discovered that there was an exponentially increasing lag between the time the license plate is recognized and when the OCR characters are recognized and printed on the terminal, this lag appears to increase more rapidly when the system is being used in real time with live camera fee. This observable effect is due to the fact that the LPR module is much faster at doing its jobs than the OCR module, the OCR is highly dependent of network speed, consequently the LPR module populates the queue a lot faster than the OCR engine depopulates it, thus the lag.



**Figure 4.3: Parallelism and concurrency-based efficiency.**

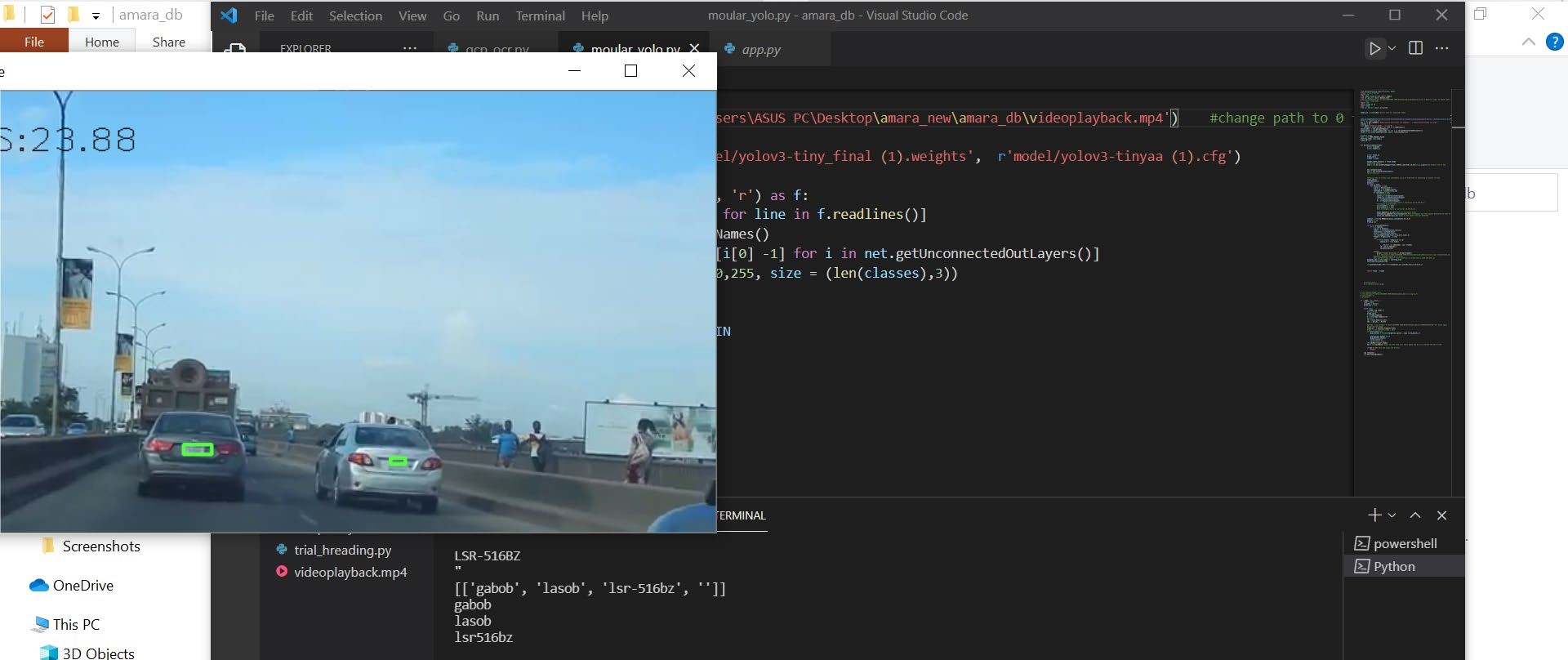
**Source: Compiled by the researcher.**

**4.4 Localization Errors**

Generally, the LPR engine does its job in localizing license plate numbers in images, however there are certain conditions which absolutely hinder the LPR engine from carrying out its functionalities, this section discusses most of this condition. Vehicle colour tone, one of the most noticeable condition which greatly affected the performance of the model was the vehicle colour, the module really struggled when it had to localize vehicles with colours similar to white, this is due to the fact that Nigerian plate numbers majorly white, this creates a scenario where the colour of the license plate blends with the colour of the car therefore making it near impossible to successfully localize the license plate ROI.

**4.5 Vehicle Tone Colour**

The colour tone of vehicles greatly affected the effectiveness of this model, in terms of plate number localisation the model performed very poorly on vehicles with light tone, this performance is highlighted using their recall and precision values.



**Figure 4.4: Illustration of wrongly localized plates.**

**Source: Compiled by the researcher.**

**4.6 Light Tone Car versus Dark Tone Car**

Recall is the measure of how many times the model predicts correctly that there is a plate number in a frame, however precision bothers about how well the model does at locating the exact location of the plate number in the image.



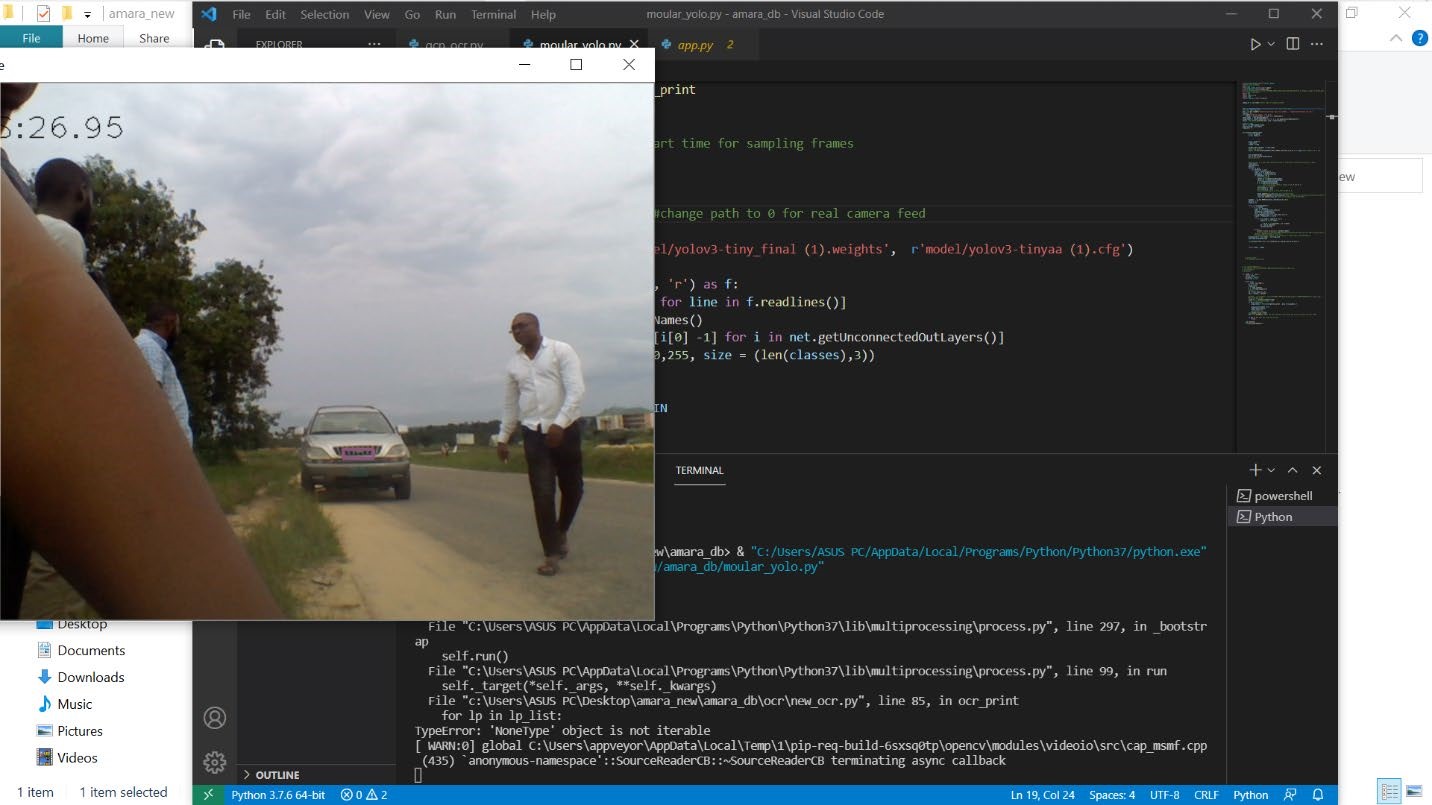
**Figure 4.5: Dark coloured vs light coloured cars.**

**Source: Compiled by the researcher.**

**4.7 Recall Versus Precision**

In dark colour vehicles, the behaviour of the precision and recall values are just typical , the precision is slightly higher than the recall value , however in light cars the precision value drops from 76 % to 25 % , however when we move to the recall value the percentage increases slightly from 70 % to 80 % ,the difference between the recall value and precision value is very big , this is due to the fact that in white cars and light tone cars the model can easily mistake any portion of the car as a license plate , since recall only considers when the model makes a correct prediction on a plate number existing in a frame ,therefore predictions made because the model mistook a portion of a light coloured car as a plate number are considered as true positives (TP), precision however can be fooled, since it bothers about how well the model does in locating the exact position of the plate number in a frame, so it knows when a model is making a right prediction for a wrong reason, thus the low precision value.

Blue coated plate numbers are very common amongst Nigerian drivers, my model failed at localizing plate numbers when they are coated, the model was tested on a silver-coloured Lexus RX330, it failed to even partially localize the plate number ROI.



**Figure 4.6: Illustration of a Correctly Localized Plate.**

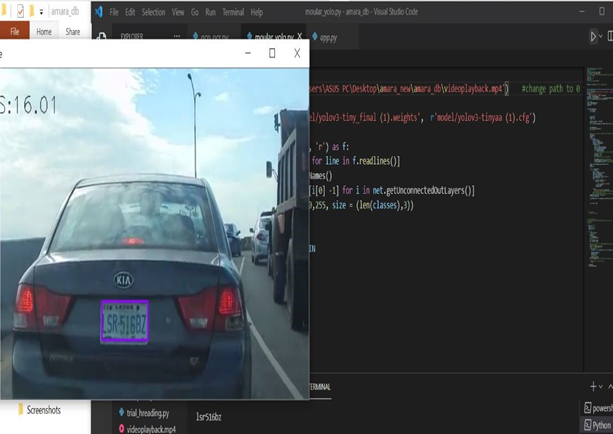
**Source: Compiled by the researcher.**

**4.8 Optical Character Recognition Engine (OCR)**

A cloud-based OCR engine was used in this work due to the high accuracy most of the models on the cloud provide, even with such high accuracy the OCR engine had some problems and performance issues.

**4.8.1 Latency**

One of the common problems with a cloud hosted OCR engine was the time it took to send a response to the cloud and receive a response after the job is completed , the total time consumed is dependent on internet speed , this latency makes it impossible for the OCR engine to synchronize with the LPR engine, this implies that when the LPR module has recognized a plate number ROI , the OCR engine takes a time (t) to extract and print out the recognized characters , this time is greatly influenced by network speed.



**Figure 4.7: Illustration of a fast accurate localization.**

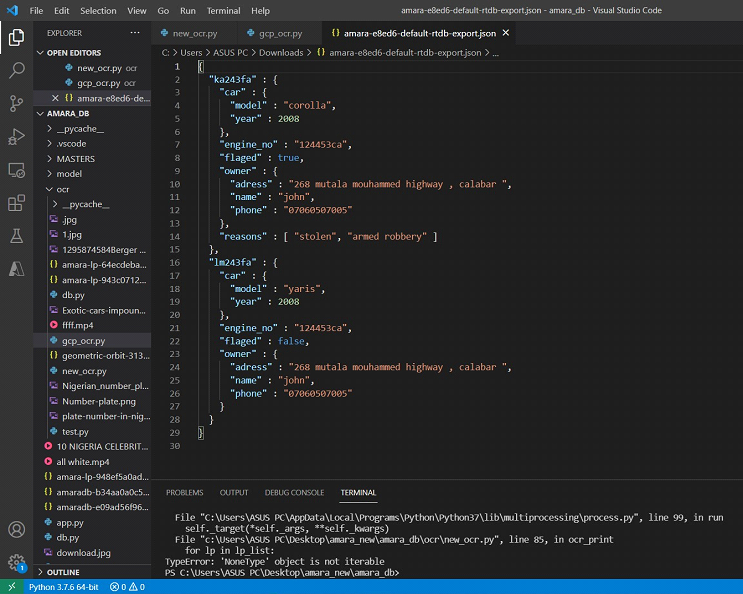
**Source: Compiled by the researcher.**

**4.8.2 Recognizing Special Characters**

The model does very poorly in recognizing special characters such as hyphen, due to this poor performance, all special characters are deleted from the plate number before it is passed to the database module for querying. The model does very poorly in recognizing special characters such as hyphen, due to this poor performance, all special characters are deleted from the plate number before it is passed to the database module for querying.

**4.9 Firebase Realtime Database**

In the database module as shown in Figure 4.8, querying was effective once a node updates a database record it almost instantly reflects on all other connected nodes, the real time database interphase worked very easy to update singular values and large records. Updating formats for single records are very easy to understand and use, the updating is done through a python in-built data structure called dictionary for single records. This data structure is really easy to understand and use. However, large updates are a bit difficult to understand and use, the database uses JSON datatype to update large records, compared to other database models. This means of updating is very easy to interpret and use, this is why most modern database frameworks like monogodb have also taken this approach to updating their database.



**Figure 4.8: Screenshot of database code.**

**Source: Compiled by the researcher.**

**4.10 Security and Authentication**

Firebase database provides two factor authentication for securing the database from unauthorized access, this approach to security in our system model, our model in deployment requires a security framework which allows all user to simultaneously update and query the database, however deleting the database record is only reserved for administrator, this form of privileged access system is exactly what the database framework offers.

**CHAPTER FIVE**

**SUMMARY, CONCLUSION AND RECOMMENDATION**

**5.1 Summary**

The existing license plate recognition systems was analysed to find a way to deploy it in the Nigerian system. This has been successfully carried out although it requires some high-end hardware to implement it efficiently. These hardware makes the frame rate a bit faster and a high-quality camera is needed to increase the accuracy of the OCR engine.

**5.2 Conclusion**

This system is effective in checking license plate. However, the environment (Nigerian transportation industry) will need some adjustments to help the system adapt and properly carry out its objectives. This improvement revolves around eliminating certain conditions which make extraction of plate numbers virtually impossible for most digital systems (this system inclusive). Also, sometimes conditions such as washed-out license plates hinder even human based system such as the one being used currently from being able to carry out the task of reading plate numbers.

For this system to be deployed effectively, all the abnormal conditions discussed in this work will need to be eliminated as these conditions hinder even law enforcement officers using the current existing license plate recognition framework from doing their jobs. Certain measures have to be taken for this system to be full-proof, some of which includes the scrapping out of blue coated license plates, enforcing that every white vehicle has a black or dark coloured bumper that holds the license plate or even adopting a totally new format for license plates which uses green colour since all shades of green can easily be recognized by computers.

With respect to the database engine, it can be concluded that there is no better alternative for a system of this nature than a real-time database. Firebase was used in this system and it performed very well as no challenges were encountered while working with it.

**5.3 Engineering Implications of Findings**

It has been realized that actualizing such a system in Nigeria is not really that far-fetched anymore. This research work has helped explore an alternative to applying it called YOLO. Also, it has contributed to engineering in Nigeria considering that right now we have a full understanding of the challenges that are faced in deploying such a system here.

**5.4 Contribution to Knowledge**

This project has succeeded in carving out a framework that could be built upon for an ALPR system that seeks to be deployed in Nigeria. This work will be pivotal towards achieving or actualizing such a dream in a minimum amount of time. This work has done a great deal of progress in finding out an efficient and practical way to deploy such a system.

**5.5 Recommendation**

For Nigerian license plates, it is advised that in the design of the plate, the Nigerian map should stand out properly to make the OCR engine a lot more accurate, as illustrated in Figure 5.1.



**Figure 5.1: Recommended License Plate Style.**

**Source: Compiled by the researcher.**

**5.6 Suggestions for Further Studies**

To design an improved version of this system, a database that contains more detailed information about vehicles like chassis number, vehicle colour, vehicle number and so on should be used. Also, future updates should have a way of automatically detecting vehicle colour and vehicle type in case if a plate number is changed, it can be crossed referenced against the colour of the vehicle and the vehicle type.

**5.7 Ethical or Legal Challenges**

Placing this system on the road could infringe on our basic human rights as people would not warmly welcome the idea of their vehicles being captured by a camera. Laws or written contracts should be put in place which ensures and limits the use of the information gathered for only the purpose it was intended for.

# REFERENCES

Abdullah, S., Hasan, M. M. and Islam, S. M. (2018). YOLO-based three-stage network for Bangla license plate recognition in Dhaka Metropolitan City. *International Conference on Bangla Speech and Language Processing*, 2(5): 1-6.

Ali, F., Rathor, H. and Akram, W. (2021). License plate recognition system. In: *2021 International Conference on Advance Computing and Innovative Technologies in Engineering,*3(4): 1053-1055.

Allam, M. (2012). Canny Edge-Detection Based Vehicle Plate Recognition. *International Journal on Signal Process., Image Process. Pattern Recognition,* 5(3): 1-8.

Anagnostopoulos, I. E. (2006). A license plate recognition algorithm for intelligent transportation applications. *IEEE Transactions on Intelligent Transportation Systems*, 7(3):377-392.

Angeline, L., Kow, W. Y., Khong, W. L., Choong, M. Y. and Teo, K. T. (2012). License plate

character recognition via signature analysis and features extraction. *International Conference*

*on Computational Intelligence Modelling and Simulation,* 4(2): 1-6.

Balaji, G. and D. Rajesh. (2017). Smart vehicle number plate detection system for different countries using an improved segmentation method. *Imperial Journal of Interdisciplinary Research (IJIR), 3*(6): 707-712.

Bharat, B. (2013). License plate recognition system using neural networks and multi thresholding technique. *International Journal of Computer Applications,* 84(5): 0975-8887.

Buhuss, E. R., Timis, D. and Apatean, A. (2016). Automatic parking access using Open ALPR on

Raspberry Pi3. 57(3): 10-15.

Chen, T. (2009). License plate text localization using DWT and neural network. *Institute of Electrical and Electronic Engineers International Conference on Granular Computing*, 2(3): 73-77.

Chowdhury, D., Mandal, S., Das, D., Banerjee, S., Shome, S. and Choudhary, D. (2019). An adaptive technique for computer vision based. *Institute of Electrical and Electronic Engineers*, 6(8): 978-983.

Dhanlakshmi, K. and Leni, E. S. (2017). Instance vehicle monitoring and tracking with internet of things using Arduino. *International Journal on Smart Sensing and Intelligent Systems*,2(6):372-380.

Dholakia, K. (2015). A survey on handwritten character recognition techniques for various Indian languages. *International Journal of Computer Applications*, *115*(1).

Du, S., Ibrahim, M., Shehata, M. and Badawy, W. (2013). Automatic license plate recognition (ALPR): A State-of -the-Art Review. *Institute of Electrical and Electronic Engineers Transactions Circuits Syst. Video Technology,* 23(2): 311-325.

Dwivedi, U., Rajput, P. and Sharma, M. K. (2017). License plate recognition system for moving vehicles using Laplacian edge detector and feature extraction*. International Journal of Engineering Research and Technology*, 407-412.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate

object detection and semantic segmentation. In *Proceedings of the Institute of Electrical and Electronic Engineers conference on computer vision and pattern recognition* 3(8): 580-587.

Gou, C., Wang, K., Yao, Y., and Li, Z. (2016). Vehicle license plate recognition based on external regions and Restricted Boltzmann Machines. *Institute of Electrical and Electronic Engineers Transactions on Intelligent Transportation Systems,* 17(4): 1096-1107.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *Computer Vision and Pattern Recognition, Institute of Electrical and Electronic Engineers* 2(5): 770-778.

Hsu, G., Chen, J., and Chung, Y. (2013). Application-oriented license plate recognition. *Institute of*

*Electrical and Electronic Engineers,* *2*(4): 552-561.

Htay, M. M., and Gopalakrishnan, A. K. (2016). Localization and recognition of a myanmar license plate based on partially cut character structure. *International Conference on Information and Communication Technology and Knowledge Engineering, 14*(2): 38-43.

Hung, K.-M., and Hsieh, C.-T. (2010). A real-time mobile vehicle license plate detection and recognition. *Tamkang Journal of Science and Engineering, 13*(4):433-442.

Jin, X., Tang, R., Liu, L., and Wu, J. (2020, September 10). Vehicle license plate recognition for fog-haze environments. *Image Processing*, 1(5):1-12.

Joshi, A. S., and Kulkarni, D. A. (2017). Automatic number plate recognition and iot based vehicle tracking. *International Research Journal of Engineering and Technology (IRJET)*, 7(5):1503-1507.

Karasu, S., Altan, A., Sarac, Z., and Hacioglu, R. (2017). Histogram based vehicle license plate recognition with KNN method. international Conference on advance Technology, 6(2): 1-4.

Khin, O., Phothisonothai, M., and Choomchuay, S. (2018). Myanmar character extraction from vehicle images using aspect ratio and bounding box. *International Workshop on Advanced Image Technology*,3(2): 1-4.

Koval, V., Turchenko, V., Kochan, V., Sachenko, A., and Markowsky, G. (2003). Smart License Plate Recognition System Based on Image Processing Using Neural Network. 123-127.

Kumthekar, A., Owhal, M. S., Supekar, M. S., & Tupe, M. B. (2018). Recognition of vehicle number plate using Raspberry pi. *International Research Journal of Engineering and Technology*, 5(4), 391-394.

Laroca, R., Severo, E., Zanlorensi, L. A., Oliveira, L. S., Gonçalves, G. R., Schwartz, W. R., and Menotti, D. (2018). A robust real-time automatic license plate recognition based on the YOLO detector. In: *2018 International Joint Conference on Neural Networks (IJCNN)* 3(8): 1-10.

Li, S., and Li, Y. (2015). A recognition algorithm for similar characters on license plates based on improved CNN. *International Conference on Computational Intelligence and Security (CIS),* 11(3): 1-4.

Lian, X., Zhang, T., and Liu, Z. (2010). A novel method on moving-objects detection based on background subtraction and three frames differencing. *Measuring Technology and Mechatronics Automation*, 8(2): 252-256.

Lin, C. and Sie, Y. (2019). Two-Stage License Plate Recognition System Using Deep Learning. *Institute of Electrical and Electronic Engineers International Conference on Applied System Invention,* 2(3):1-4.

Lin, C. H. and Wu, C. H. (2019). A lightweight, high-performance multi-angle license plate recognition model. In *Institute of Electrical and Electronic Engineers International Conference on Advanced Mechatronic Systems,* 5(4):235-240.

Lin, C., Lin, Y., and Liu, W. (2018). An efficient license plate recognition system using convolutional neural networks. *Institute of Electrical and Electronic Engineers International Conference on Applied System Invention*, 7(3): 224-227.

Lin, T., Dollar, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). feature pyramid networks for object detection. *Institute of Electrical and Electronic Engineers Computer Vision and Pattern Recognition,* 8(4): 936-944.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Institute of Electrical and Electronic Engineers Computer Vision and Pattern Recognition,* 7(5): 3431-3440.

Ma, X., Pan, R. and Wang, L. (2010). License plate character recognition based on Gaussian Hermite moments. In:  *Institute of Electrical and Electronic Engineers Second International Workshop on Education Technology and Computer Science*, 2(3): pp. 11-14.

Mantas, J. (1986). An overview of character recognition methodologies*, Pattern Recognition,* 3(5):4-8.

Nguyen, T. T. and Nguyen, T. T. (2012). A real time license plate detection system based on boosting learning algorithm. *International Congress on Image and Signal Processing*, 7(3): 819-823.

Nukano, T., Fukumi, M., and Khalid, M. (2004). Vehicle license plate character recognition by neural networks. *International Symposium on Intelligent Signal Processing and Communication Systems*, 2(7): 71-75.

Panahi, R. and Gholampour, I. (2016). Accurate detection and recognition of dirty vehicle plate numbers for high-speed applications. *Institute of Electrical and Electronic Engineers Transactions on intelligent transportation systems*, 18(4), 767-779.

Rani, L. P. J., Kumar, M. K., Naresh, K. S., and Vignesh, S. (2017). Dynamic traffic management system using infrared (IR) and Internet of Things (IoT). In: *2017 Third International Conference on Science Technology Engineering and Management,* *Institute of Electrical and Electronic Engineers,* 8(4): 353-357.

Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *Institute of Electrical and Electronic Engineers Transactions on Pattern Analysis and Machine Intelligence, 39*(6): 1137-1149.

Sagar, G. k. (2017). Real time implementation of optical character recognition. *International Journals of Advanced Research in Computer Science and Software Engineering*, 7(2): 149-156.

Sahu R.M., B.Gaikwad, N., B.Sandage, C. and S.Shinde, V. (2017). IoT based automatic vehicle license plate recognition system. *International Journal of Advanced Research, Ideas and Innovations in Technology*, 3(2): 3552-3558.

Sajid, H., and Cheung, S. C. (2015). Background subtraction for static and moving camera. *Institute of Electrical and Electronic Engineers, International Conference in Image Processing (ICIP)*, 9(10): 4530-4534.

Sajjad, K. M. (2010). Automatic license plate recognition using python and OpenCV. *Department of Computer Science and Engineering MES College of Engineering*.

Silva, S. M. and Jung, C. R. (2020). Real-time license plate detection and recognition using deep convolutional neural networks. *Journal of Visual Communication and Image Representation*, *71*: 102773.

Tejas, B., Omkar, D., Rutuja, D., Prajakta, K., and Bhakti, P. (2017). Number plate recognition and document verification using feature extraction OCR algorithm. In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, 8:1317-1320.

Wang, J., Zhou, W., Xue, J., and Liu, X. (2010). The research and realization of vehicle license plate character segmentation and recognition technology. *International Conference on Wavelet Analysis and Pattern Recognition,* 5:101-104.

Wen, Y., Lu, Y., Yan, J., Zhou, Z., Deneen, K. M., and Shi, P. (2011). An algorithm for license plate recognition applied to intelligent transportation system. *Institute of Electrical and Electronic Engineers, Transactions on Intelligent Transportation Systems, 12*(3): 830-845.

Wu, S., Zhai, W. and Cao, Y. (2019). PixTextGAN: structure aware text image synthesis for license plate recognition. *Institution of Engineering and Technology, Image Processing*, 13(14): 2744-2752.

Yilmaz, K. (2011). A smart hybrid license plate recognition system based on image processing using neural network and image correlation. *Innovations in Intelligent Systems and Applications*, 5:30-36.

Yu, M., and Kim, Y. (2000). An approach to Korean license plate recognition based on vertical edge matching. *Institution of Engineering and Technology, Image Processing*, 3: 1-6.

Zhai, X., Bensaali, F., and McDonald-Maier. (2013). Automatic Number Plate Recognition on FPGA. *International Conference on Electronics, Circuits and Systems (ICECS)*, 325-328.

Zhuang, H., Low, K. S., and Yau, W. Y. (2012). multichannel pulse-coupled-neural-network-based color image segmentation for object detection. *Institution of Engineering and Technology, Image Processing, Transactions on Industrial Electronics*, 9: 3299-3308.