**Computer Vision-Based** **Vehicle Integrated**

**Information Collection System**



Shangyuan Liu

25344136

[25344136@students.lincoln.ac.uk](mailto:25344136@students.lincoln.ac.uk)

School of Computer Science

College of Science

University of Lincoln

Submitted in partial fulfillment of the requirements for the

Degree of BSc(Hons) Computer Science

*Supervisor:* Dr. Shouyong Jiang

April 2021

**Acknowledgements**

Firstly, I want to thank somebody, and somebody else. Here is another thing.

**Abstract**

摘要在其整体上呈现了项目的简要摘要，并用来帮助读者快速确定项目的目的，上下文和结果。 摘要的目的是让读者在不需要阅读整个文档的情况下就能了解报告的内容，并帮助标记报告的内容。它还通常包含项目的关键结果是什么的简要说明。

Vehicle detection and vehicle license plate recognition (VLPR) is an indispensable component of an intelligent traffic management system, with the quick development of the information technique. Therefore, this project's purpose is to investigate and analyse the various fundamental theories about VLPR technology, which will be used in developing a computer vision-based vehicle integrated information collection system. The issue of accuracy and validity in complex light environments, low resolution, and random distribution has received considerable critical attention.

**Table of Contents**

[Introduction 1](#_Toc58407584)

[1.1 Some notes 1](#_Toc58407585)

[1.2 Testing some mathematics 1](#_Toc58407586)

[1.3 Undergraduate Project Report 1](#_Toc58407587)

[1.4 Referencing 1](#_Toc58407588)

[1.4.1 Ludography 2](#_Toc58407589)

[Literature Review 3](#_Toc58407590)

[2.1 Background 3](#_Toc58407591)

[2.2 Related Literature 3](#_Toc58407592)

[Methodology 4](#_Toc58407593)

[3.1 Project Management 5](#_Toc58407594)

[3.2 Software Development 5](#_Toc58407595)

[3.3 Toolsets and Machine Environments 5](#_Toc58407596)

[3.4 Research Methods 6](#_Toc58407597)

[Design, Development and Evaluation 7](#_Toc58407598)

[4.1 Software Development Projects 7](#_Toc58407599)

[4.2 Research Projects 8](#_Toc58407600)

[Conclusions 9](#_Toc58407601)

[Reflective Analysis 10](#_Toc58407602)

[References 11](#_Toc58407603)

**List of Figures**

3.1 A picture of the Brayford from Google Images. . . . . . . . . . . . . . 5

**List of Tables**

1. Here is a table. . . . . . . . . . . . 5

**Chapter 1**

# Introduction

## 1.1 Some notes

It is worth noting that this document is a project report template for the University of Lincoln, School of Computer Science. It should give you some direction and instruction for formatting and presenting your project report. If you have any suggestions or issues, please contact mdoughty@lincoln.ac.uk. It has been derived from the Latex PDF however, so there might be some issues – but ones I suspect you can overcome!

## 1.2 Testing some mathematics

Here are two equations using the equation editor (1, 2):

(1)

(2)

And here is some text with some nice inline maths, (*x,y*) wow *γ* so cool *ρ*.

## 1.3 Undergraduate Project Report

Currently, this template is set up for use with undergraduate project reports. However, the template can be modified fairly easily to conform to, for example, a MComp project report.

**Chapter 2**

# Literature Review

## 2.1 Background

In recent years, intelligent vehicle information monitoring systems have played a positive role in the management and safety of society. In addition, with the rapid development of computer vision technology, some cutting-edge applications in the automotive field have gained widespread popularity, such as VLPR, vehicle tracking, and autonomous driving. Take the VLPR system as an example, the previously published research on this field is circumscribed and problematic, with most of the literature focusing on the use of image processing and Optical Character Recognition (OCR) and other traditional methods. License plate positioning is a prerequisite for the implementation of license plate character recognition, however, Maglad’s (2012) analysis does not take account of the issue of license plate types adequately, nor does he examines how to accurately locate and segment the license plate area under the conditions of poor lighting background, blurred photos, and skewed angles. It is necessary to emphasize that the accuracy of OCR technology depends on the quality and quantity of the original character template in the database. As well as that, the identification speed of OCR should be improved as a real-time system.

Fig. 1: The license plate images under bad circumstances from Google



As is shown in the sample images Fig. 1, where it can be seen the negative influences of the above threats clearly. Therefore, I intended to locate the license plate, identify the license number, and other valid information more accurately via a new approach, which is combining convolutional neural network (CNN), U-Net (image segmentation), and OpenCV (cv2) technical etc. - to implement high-precision recognition of license plate and vehicle information such as plate number, brand, model, colour. In comparison to the traditional method of image processing that the proposed improvement approach (U-Net, CNN) for the VLPR system is accurate and robust. (Chowdhury et al., 2020). So in the following project work, it is necessary to compare the traditional identification methods with the new improved methods and to draw conclusions.

## 2.2 Related Literature

This section presents a detailed overview of the literature that relates to this project. The literature reviews will be conducted in two thematic directions, which are vehicle colour recognition, license plate location and recognition respectively. Due to the project focused features, this section will place emphasis on the second thematic direction.

### 2.2.1 Vehicle Colour Recognition

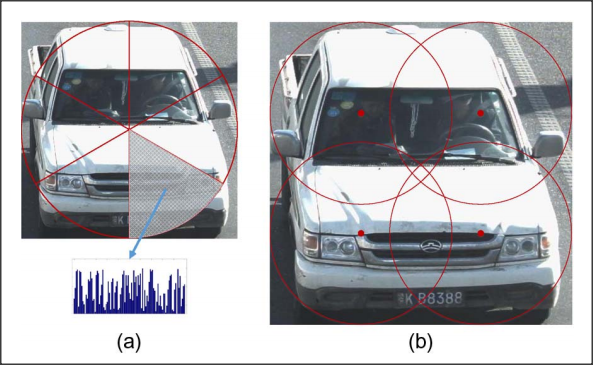
The traditional approach is to convert the image in RGB colour space to HSV space and determine which colour family the car in a given image most likely belongs to base on the Hue, Saturation, and Value (H, S, V) ranges of the nine common colours. [In](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [allusion](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [to](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) this question, Chen et al. (2014) has adopted a different approach to solve this issue based on the framework of Bag-of-Word (BoW) method, which has been put forward by [Zellig Harris](https://en.wikipedia.org/wiki/Zellig_Harris) (1954). Chen proposed methods is to recognize the dominant colour of vehicles via the region of interest (ROI) and separate the objects into subregions, and generate a histogram for each subregion, and use a linear SVM model to learn it. As can be seen from the Fig. 2, the feature context (FC) method divides the vehicle image into a number of sector sub-regions.

Fig. 2. FC Method to process the image (Pan Chen)

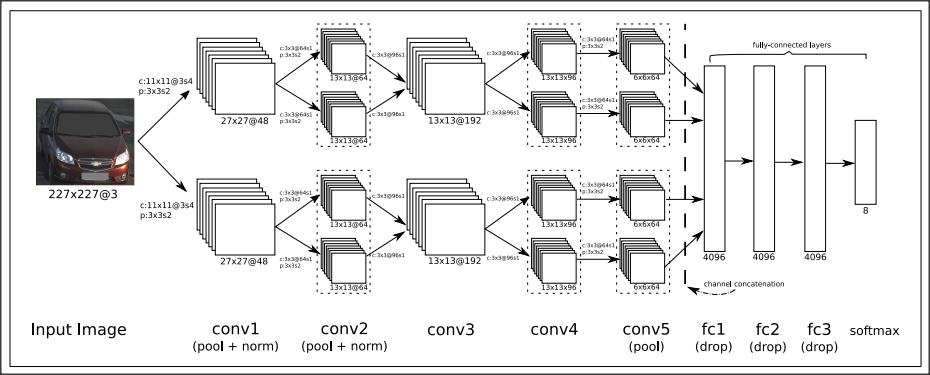
The research of Chen has come up with a good solution, however, several practical questions arise when dealing with multiple vehicles in one image. Particularly, the image gets excessively divided as the amount of subregions expands, so the discriminative capacity of the descriptor decreases. As a consequence, Rachmadi and Purnama (2018) had improved the vehicle colour recognition project based on Chen's research. They suggest a CNN-based method (the architecture image can be seen in Fig. 3) to recognize vehicle colour, the final recognition accuracy was 94.47%, and the CNN model's predictions were 2% more accurate than the results of Chen's study.

Fig. 3: The architecture of CNN (Reza Fuad Rachmadi)

### 2.2.2 License Plate Location and Recognition

It is fair to say that the VLPR technology has developed along with the maturity of artificial intelligence technology, and is a separate branch from the OCR system. In terms of OCR, it was invented in the 1950s and has been criticized and ignored because of its low recognition accuracy and slow speed. And Islam, N (2016) indicated that “the earliest OCR systems were not computers but mechanical devices that were able to recognize characters, but very slow speed and low accuracy”.

This system became commercially available in the late 1990s due to the demands of traffic management. And a systematic review of prospective observational studies found that the result offered by Jun-Wei Hsieh (2002) in [Morphology-based License Plate Detection from Complex Scenes] demonstrate that the robust approach to this issue has been proposed. And there was a great effort has been devoted to the study of multiple morphological operations, which used to find high-contrast areas as important features for license plate detection. Also,

V. Kocal (2003) indicated that using image fusion, neural networks, and iterative thresholding operation to identify license plate of vehicle. However, most of the previous studies do not take into account the future social demand and the rapid development of the automotive industry. Due to limited by the capabilities of conventional algorithms, and the low level of camera imaging, etc., so VLPR was not ready for widespread use compared to the current stage of technology. Therefore, it is concluded that these approaches may not be practical in all situations by means of literature review.

During the past twenty years, [many](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [investigations](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) have appeared in recent years documenting VLPR with the rise and technical maturity of machine learning and deep learning techniques. As a result, the previous research has demonstrated an algorithm that license plate recognition using a binary time-delay neural network (TDNN), which is a classification used for training the neural network. The TDNN architecture diagram is shown in Figure 4. According to the research from Ghasrodashti and Yazdi (2020) has shown that “hard conditions such as the distance and angle variations as well as weather and light conditions are considered, and the recognition rate obtained by the proposed algorithm was 70%.” Moreover, in order to prove that the algorithm can contribute to better results, the proposed algorithm (TDNN) is also compared with the previous method (SVM).

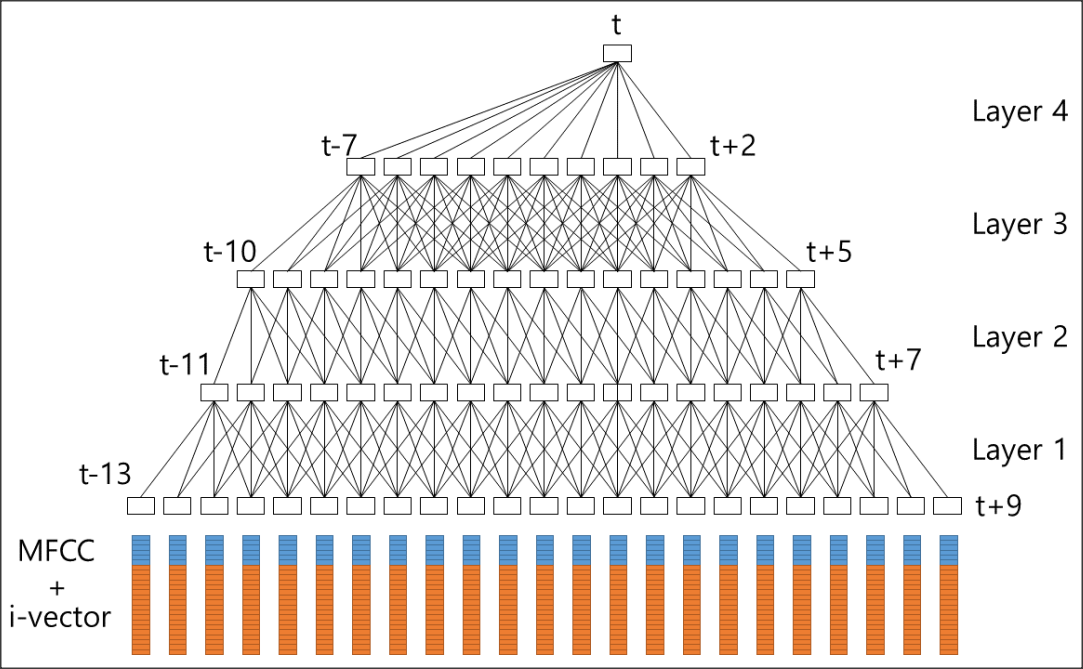


Fig. 4: TDNN architecture (Park, H., Lee, D., Lim, M., Kang)

At present, more research is invested in deep learning in the field of VLPR, especially the end-to-end training network method is extensively used. Glasmachers (2017) has shown that “it is trained in a holistic manner based on a single principle”. The overall architecture of the proposed end-to-end principle is shown in Fig. 5. And it approaches complicated problems by using the Deep Neural Networks (DNNs) model, which is made up of many layers. To end this, Qin and Liu (2020) proposed a strategy for VLPR, which is based on deep neural networks and end-to-end methods. And according to the research, it can decrease the error propagation effectively between the models of location and recognition network by means of common features, convolutional neural network, and multi-task learning strategy. Therefore, this basic structure and theoretical knowledge can be used as the basis for this project.

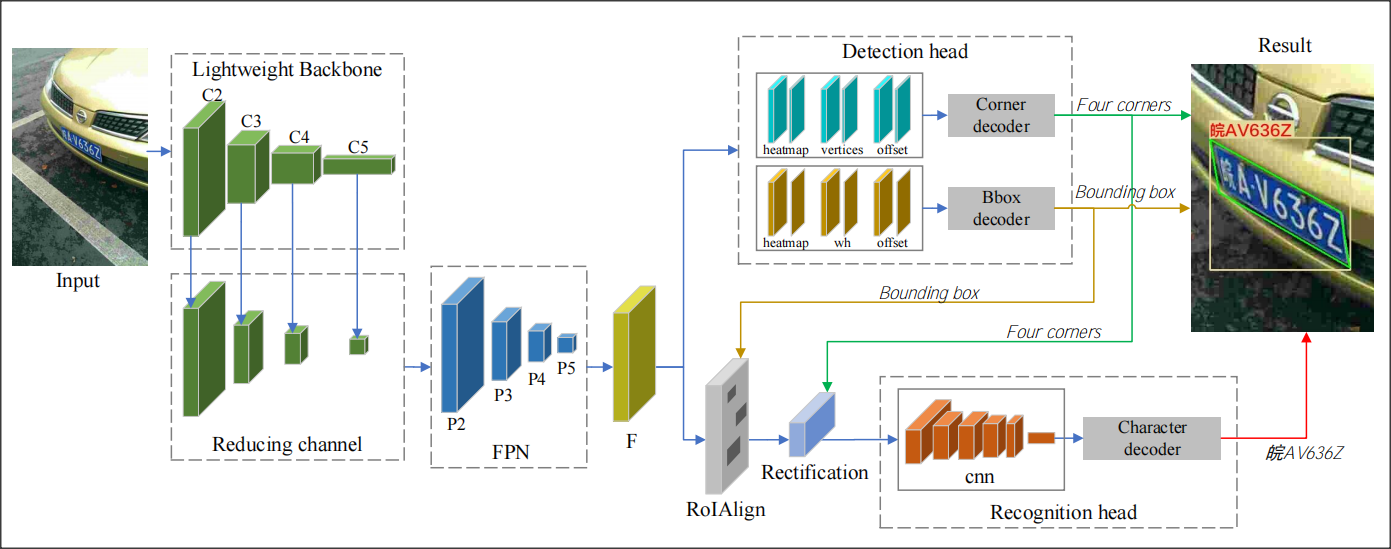


Fig. 5 The end-to-end network structure model for license plate location and recognition (Qin)

However, the end-to-end method is still under debate, and I believe it is not suitable and circumscribed in some cases. This discrepancy could be attributed to the following several reasons. Firstly, a large number of data samples are required to train the model. Secondly, it is difficult to optimize or modify the model system because the DNN model parameters must be replaced and retrained. Moreover, the method is difficult to interpret and validate due to the complicated neural network architecture, which point is significant for the automotive industry particularly. Therefore, the study by Glasmachers (2017) provided that “We have demonstrated that end-to-end learning can be very inefficient for training neural network models composed of multiple non-trivial modules. End-to-end learning can even break down entirely; in the worst-case none of the modules manages to learn. In contrast, each module is able to learn if the other modules are already trained and their weights frozen.”

To sum up, the technologies in this area generally concerns themselves with image processing and computer vision, also it can be divided in to more directions for development. [Although](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) some results would seem to suggest that the current VLPR technology has some limitations, it is against this background that license plate recognition technology, which focuses on non-traffic areas, is gaining more and more attention.

**Chapter 3**

# Methodology

## 3.1 Project Management

Projects are vital to modern society, to companies and to individuals. Project management is the application of resources to objectives in order to achieve the goals of a project and to meet the established requirements of all parties. Therefore, a suitable project management methodology can provide an effective assurance for the implementation of the project. And there are several common project management methodologies such as Agile, Scrum, Waterfall, Kanban, Extreme Programming (XP), and Lean etc.

Table...

|  |  |
| --- | --- |
| **The Natures of this Project** | |
| **Name** | Computer Vision-Based Vehicle Integrated Information Collection System. |
| **Relativity** | The project is structured on the basis of the 'Project Course' assessment, and the students' tasks constitute the project. |
| **Temporary** | The project starts on 12 October 2020 and ends on 13 May 2021, for a total project duration of approximately 31 weeks. When the project ends, the products of project will remain and be used. |
| **Purpose** | Complete functions such as model recognition, color recognition, license plate recognition and vehicle counting with high accuracy. |
| **Constraint** | The knowledge and techniques that I currently mastered are too superficial to complete the project, and I need to continue learning. And a low configuration environment of the computer wastes lots of time in the development process. |
| **Innovation** | During the development of this project, new algorithms will be used to perform functions such as license plate recognition. But in the process of innovation, There will be uncertainties, which can make the project in risk. |

[As](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [shown](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [in](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [the](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) Fig. , a complete project management process should consist of five phases, which are project initiation, planning, execution, monitoring and closure respectively. And in managing this project, the members of the project should shoulder their own responsibilities, have different skills and experience accumulation, and arrange the work correctly and reasonably. In addition, throughout the project members can learn from the project, whether from technical aspects as well as organisational and management aspects.



Fig. 5 Phases of Project Management (by [Kate Eby](https://www.smartsheet.com/content-center/author/Kate Eby) on May 29, 2018)

There are diversified types of management method that have been applied in the area of project management, and Toljaga-Nikolic et al. (2017) believe that “the project management approaches should be prepared to deal with requirements of different level of completeness and changes during the project life cycle.” Therefore, in the case of this project, agile project management methodologies will be used. Agile is an emerging project management model that simplifies the cumbersome processes and documentation of traditional project management. When requirements are not clear, usable software can be developed in a shorter cycle to help customers describe their requirements. And through multiple iterations of requirements to enrich the product features of the project.

The main purpose of this project is to develop a VLPR system, so plate recognition function is have priority rather than the others function. However, other functional requirements and specific algorithms are not well defined, so in order to reduce the risk of requirements, continuous analysis, development and evaluation are required to complete the project. Overall, the product of this project should have good functionality, a high-efficiency algorithm model and high recognition accuracy.

Using the agile methodology, the core functionality is analysed and developed at first, and then the unclear requirements are developed iteratively once the main functionality has been implemented. And there is an evidence to support this point that “when the project goal is clear but the solution is missing some or most parts, then an agile approach should be applied in the project management.” Toljaga-Nikolic et al. (2017). As a consequence, agile method is very interactive and the whole project can be adjusted quickly that is suitable for this project.

Table...

|  |  |
| --- | --- |
| **Advantages and Disadvantages of Agile Method** | |
| **Advantages** | **Disadvantages** |
| **Risk Reduction:** Agile methods are an iterative, step-by-step approach that can predicts the success or failure of a project early, so the PMO can take corrective measures in time before project failure.  At least part of the project product is always available, so the project never ends up as a complete failure. | **Rely on Customer Interaction:** Since the whole project is designed according to the requirements provided by the clients, the agile approach is focused on customer participation. As a result, if they do not understand the product features, the process of production will go out of track. |
| **Predictable Results:** By understanding the productivity of developers, the development team can help the PMO to accurately predict the progress and outcome of the project. | **No Specific Goals:** It is easy to get off-topic and keep adding new functions during subsequent iterations. This means that there is no clear endpoint to project. |
| **Improving Product Quality:**  Users can modify their requirements as business needs change to establish a more suitable model.  Project customers, developers, product managers and Scrum leaders can monitor the progress of the project. | **Lack of Documentation Support:**  If there is a high turnover of project staff, this can make it difficult to maintain the project.  The sustainability, maintainability and scalability of the project are at greater risk. |

For the agile method, there is a [conclusion](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) demonstrate that “in the comparative study of agile software development with other software development models we conclude that agile project is much better than other software development process in terms of productivity, performance, faster time cycles, risk analysis.” (Sharma et al., 2012). Although the agile methodology has many advantages in some project management, it also exists with some disadvantages. Table 1 provides a detailed analysis of the advantages and disadvantages of the agile method.

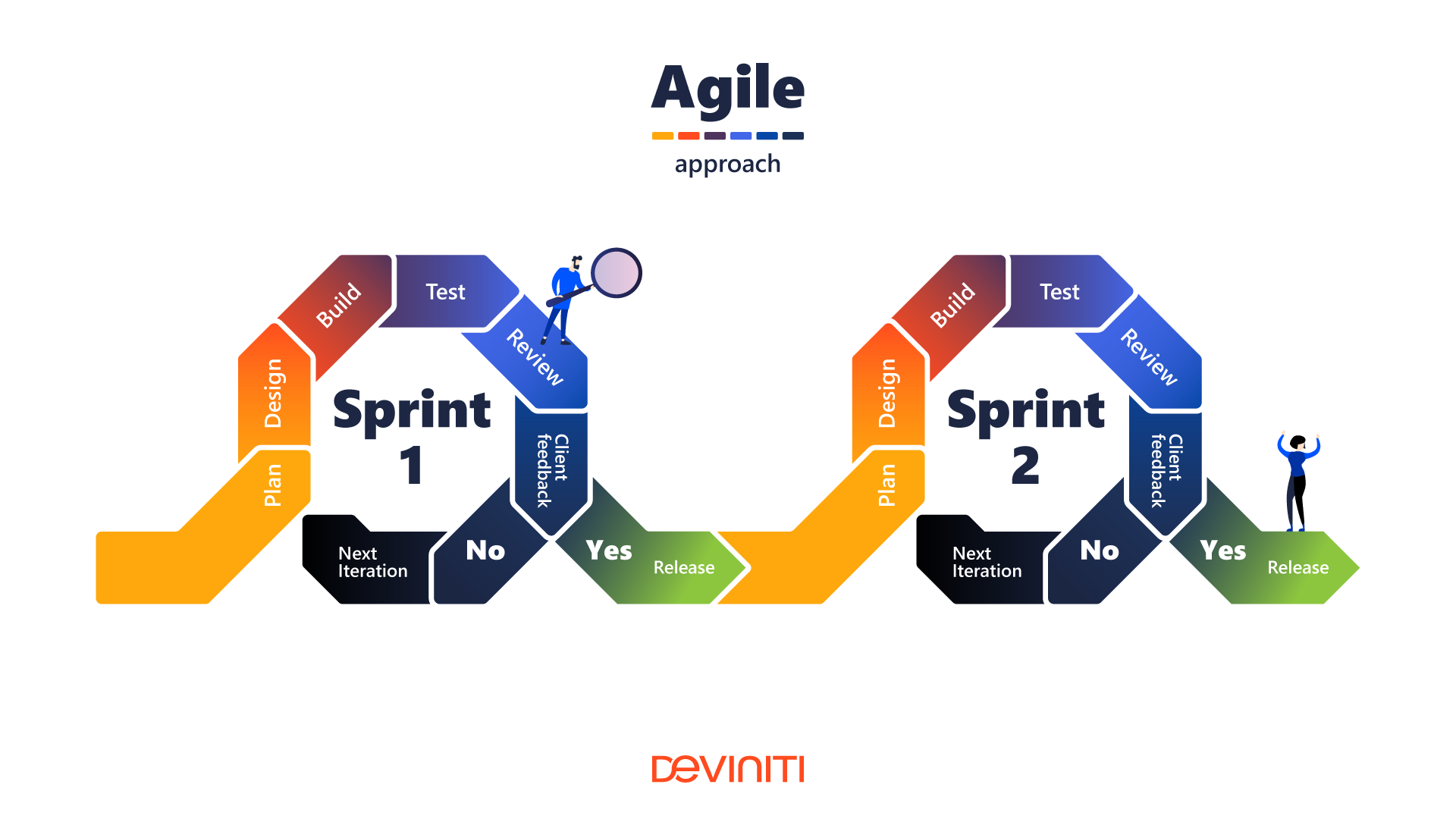


Fig. ....Agile Project Management Structure

https://blog.deviniti.com/agile-devops/requirements-documentation-project-management-approach/

However, taking full advantage of the benefits of agile development can minimise the negative effects it can have. This management method can provide a good solution to a project when the project's objectives are clear, the client is able to actively participate in the project and the project's activities are highly iterative. (Toljaga-Nikolic et al., 2017). Fig. shows the structure diagram of the agile approach, where the project becomes more complete and mature after many iterations.

Basically, using project management software is a good solution to increase productivity. In this project, GIT will be used as a project management tool and the content and results of the project will be uploaded to the GitHub platform. On GitHub, developers can coordinate, track and update the project's work progress, thus keeping the project transparent and on schedule.

## 3.2 Software Development

During the development stage, the software will be developed using an iterative and incremental approach. Because the core of the system's functionality is to implement accurate licence plate recognition, it is possible to carry out requirements analysis, design, implementation and testing operations several times on the basis of the core functionality. This method has a higher development efficiency and success rate in the software development process. And it is significant that the Multiple iterations of the software development cycle can be in operation at the same time during software development.

In order to obtain the functionality of the system and the technical approach used, a detailed analysis of the relevant project, technical and functional requirements will be carried out using data study and survey methodology. The product of this project does not have many functional requirements and it should be focused on algorithmic research, so the questionnaires and interviews approach is not suitable for the development of this project. Instead, the approaches allow for an understanding of the application scenarios, functional requirements and the latest technologies used in the project from the relevant literature. Moreover, the methods are academic and efficient because it targets academic journal articles or official data. The survey findings are evaluated to come up with comprehensive and descriptive results. (Wohlin et al., 2006).

## 3.3 Tool-sets and Machine Environments

### 3.3.1 Tools-Sets

In this chapter, an overview and evaluation of the various development tools and software used in the development and management of this project are presented. In terms of project development, I chose Python as the programming language and Visual Studio Code (VS Code) is the integrated development environment (IDE).

**Python (3.9.2):** The vehicle integrated information collection system project is based on computer vision (CV) technology, and techniques from other disciplines related to CV are also used in the project, such as image processing, machine learning, deep learning, etc. The main programming languages currently used in the computer vision area include C++, and Python, they are compared and analyzed in Table 1. The benefit of developing in Python is that it has an excellent Python developer community, where developers can find a wealth of content about these libraries with a simple search. (Sarkar et al., 2018). Therefore, the Python 3.9.2 was finally chosen to develop the project.

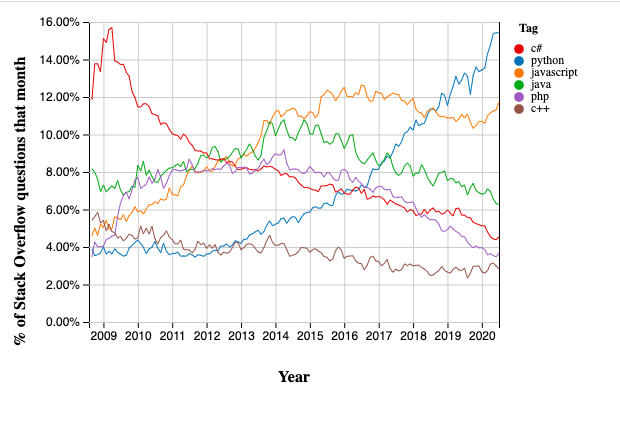


Table 1: The latest annual ranking of popular programming languages by IEEE Spectrum

Table....

|  |  |
| --- | --- |
| **The Comparison of Programming Languages** | |
| **Language** | **Summary of each Programming Language** |
| **Python** | As can be seen from Figure 1, Python is currently the most used and popular programming language, also in the field of machine learning and deep learning, it is currently the most widely used language, so there are many open source projects and courses for learning. Especially for image processing, OpenCV supports a Python interface; for scientific computing, Anaconda also provides a rich package interface. On the other hand, Python has a powerful and effective machine learning library - scikit-learn, and exhaustive resource (numpy, matplotlib).  However, Python is not suitable for embedded development because of inefficient execution. When using a large dataset to train a deep network model, the running speed is very slow and takes up too much memory for too long to do other work. |
| **C++** | In the field of vision and image processing, C++ favours underlying development, so suitable for larger projects and code execution with high efficiency. C++ is similar to Python, also supports interfaces to third-party tools such as OpenGL、EmguCV、OpenCV etc. But the inefficient development that is only suitable for commercial and industrial projects, does not apply to the development of this project. |

**Visual Studio Code:** Python has a variety of integrated development environment (IDE) for professional development, such as PyCharm, Visual Studio Code, and Vim etc. It is indicated in Table 2. That the differences between each software. According to the current computer configuration and project requirements, I have chosen VS Code as the IDE for my project development.

Table....

|  |  |
| --- | --- |
| **The Comparison of Integrated Development Environment** | |
| **Language** | **Summary of each IDE** |
| **PyCharm** | For PyCharm IDE, one of the most widely used IDEs for the Python programming language. It has many functions to help users improve their productivity when developing in the Python language. However, for a more full-featured IDE and to support web development you need to use the professional version of Pycharm, which is available for a fee. For the downside, PyCharm takes up a lot of RAM on your computer and is not ideal for low configuration computers. |
| **VS Code** | T[he](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [other](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [one](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [is](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) VS Code, which is a free open source IDE created by Microsoft for development in Python and other programming languages, with a rich ecosystem of plugins that can be easily searched for, installed and managed directly in the VS Code editor. Moreover, VS Code is more open source than sublime, faster than the Atom and more lightweight than Webstorm. |

**OpenCV (4.5.1):** A significant cross-platform computer vision library- OpenCV (Open Source Computer Vision Library) was also used in the development process. It can be used to develop real-time image processing, computer vision and pattern recognition programs because a variety of basic image processing functions and vision algorithms are encapsulated in the library. More importantly, OpenCV is open source and provides an interface to Python for project development efficiently.

**Tensorflow (2.5.0):** It is quite important to choose suitable Python Libraries for deep learning project because a suitable framework can play a positive role in the project. At [present](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;), there are five most popular machine learning libraries, which respectively are TensorFlow, Keras, PyTorch, Caffe2, Scikit-learn. The two most mainstream frameworks, TensorFlow and PyTorch are compared and analysed in Table. In TensorFlow 2.0, Keras is the official high-level API for TensorFlow, allowing for fast and easy model design and training. And the Caffe2 incorporates all of its code into the PyTorch framework.

Although PyTorch is easier for beginners to learn and master, and I have never used any machine learning framework before, I have chosen TensorFlow as the machine learning framework for this project because it supports not only deep learning but also reinforcement learning and other algorithms. For the license plate recognition project, “deep neural networks have achieved breakthrough performance on computer vision tasks such as recognizing objects in photographs, and these tasks are a key application for TensorFlow at Google” (Abadi et al., 2016). In addition, it has complete official documents and learning materials to better support the development of this project.

|  |  |
| --- | --- |
| **The Comparison of Machine Learning Frameworks** | |
| **Language** | **Summary of each framework** |
| **TensorFlow** | TensorFlow is an end-to-end open-source platform for machine learning. It is clear from Fig. That TensorFlow is the most widely used machine learning framework. Because it plays a vital role in the fields of deep neural networks, natural language processing (NLP), image, text, and speech recognition. However, it also has some drawbacks, since TensorFlow computational graphs are written in Python, large neural network models need to take a long time to compile. |
| **PyTorch** | PyTorch is a rival to TensorFlow that was created by Facebook AI Research. It has the advantage of supporting dynamic computational graph models, also known as define-by-run. Also PyTorch is easy to learn because of the large number of pre-trained models. |

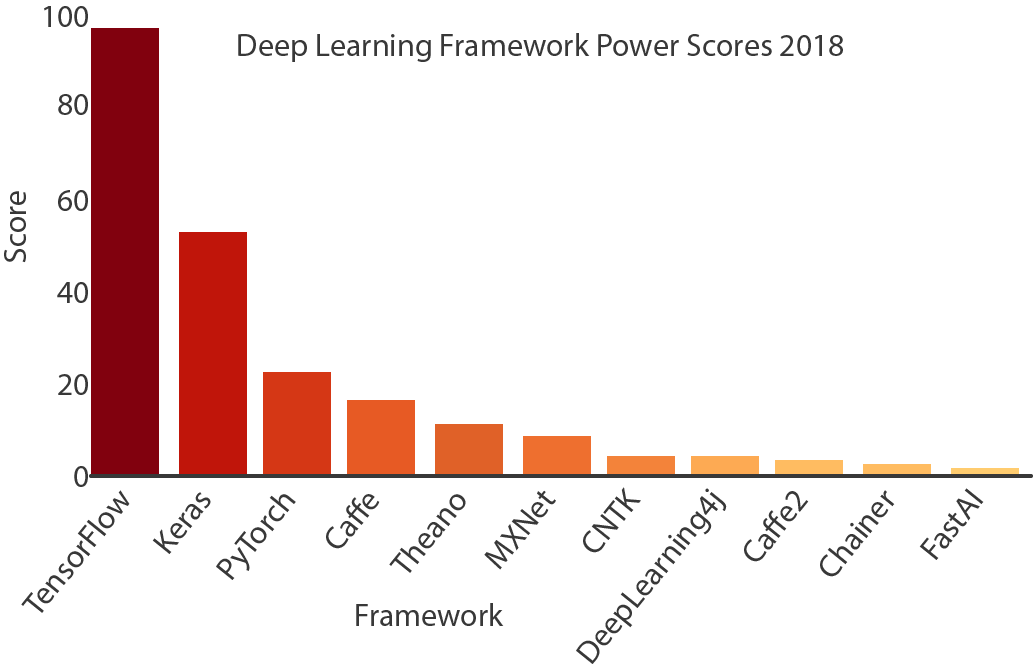


Fig. Top 5 Deep Learning Frameworks to Watch in 2018 - Towards data science

**Labelme:** Labelme is an image annotation tool developed by Massachusetts Institute of Technology (MIT), which can be used to create custom annotation tasks or perform image annotation. The main use in this project is for labelling semantic segmentation datasets of license plates. The image data is annotated in the project using lableme to generate the .json file corresponding to the image. And the reason why I chose this is that can execute the built-in function labelme\_json\_to\_dataset under Labelme to batch convert images to the corresponding x\_json file.

**Project Management - GIT:** GIT is a free and open source distributed version control system (VCS), which can handle software project management efficiently. A similar VCS to GIT is Apache Subversion (SVN), the main difference being that GIT is distributed, while SVN is not. And I selected GIT to manage the development of the project for several reasons. Blischak et al (2016) also indicated the same point that “By tracking your code development with a VCS and hosting it online, you are performing science that is more transparent, reproducible, and open to collaboration”.

* Repository localisation, offline commit support, relative independence without impacting collaborative development.
* GIT only generates a .git directory for each project, where all the project's version control information is located, unlike SVN which generates .svn directories under each directory.
* Supports quick branch switching for easy merging, and merges files faster than SVN.
* GIT's content storage uses the SHA-1 hash algorithm. This ensures the integrity of the code content.
* The ability to upload project code remotely to the GitHub platform, allowing efficient development in different environments and platforms

### 3.3.2 Machine Environments

The VLPR system has various application scenarios that mainly used in traffic monitoring, smart parking lots, vehicle access management, and toll station to recognize the license number. The development machine environment required for the system varies depending on the user requirements and application scenarios. Usually the VLPR system is used as a GUI on a Windows PC, but it can also be used for embedded development in a Linux environment.

The project will be developed on Windows 10 operation system, which supports many IDEs, machine learning frameworks and computer vision libraries, among other development tools. This makes it easy to get started with the software development process and does not require the installation of a Linux system or the configuration of a development environment. However, this does not mean that developing on Windows is better than on Linux. Most of the open-source AI platforms are built on Linux, and Python can perform at its best on Linux because of the efficiency of parallel computing and resource pool management on Linux systems, which can be adapted to different production environments.

Table ...

|  |  |  |
| --- | --- | --- |
| **Comparison of Development Machine Environments** | | |
|  | **Windows** | **Linux** |
| **Development Environment** | Windows 10 - 64 bit  CPU: i5-8300  GPU: 4 GB  RAM: 16 GB | Linux: 5.4.0-31-generic  Ubuntu 20.04 LTS - 64 bit  CPU: i5-8300  GPU: 4 GB  RAM: 16 GB |

## 3.4 Research Methods

### 3.4.1 The Aim of this Research

In terms of vehicle information collection system, the conventional identification methods do not satisfy the demands of today's society and therefore the latest technologies and algorithms have to be implemented in this system to achieve efficient, rapid and highly accurate identification functions. Therefore, the research for this project is to compare the advantages and disadvantages of different algorithms, to analyse their technical principles and finally to complete the research project based on computer vision techniques.

### 3.4.2 The Methodology of this Research

In software engineering research, quantitative analysis methods are usually more widespread, while qualitative analysis methods are often used in combination with other methods. However, in this research project, the qualitative analysis method will be used in the course of the research, and qualitative analysis is a method of inferring the nature and trend of things based on the subjective judgment and analytical ability.

Since qualitative analysis is a subjective description in written language, it is not applicable in large-scale social research. And the previous research by DeFranco and Laplante (2017) has demonstrated that “most of those papers did not follow a systematic process during the qualitative analysis. This finding is concerning as this deficiency in research analysis procedure may reduce the validity and/or completeness of the qualitative results”. Therefore, in order to improve the validity and reliability of this research, data collection and analysis should be carried out with purposes, directions and plans.

### 3.4.3 Collecting and Analyzing Research Data

In the case of qualitative research, there are various approaches to collecting data and information. In this research project, the literary research method will be used as a specific research method for qualitative analysis. It is efficient that collect subjective, self-reported data and determine questions about qualitative research through published academic journal articles, research papers, theses and books, etc. There are 20 referenced papers in this research, mainly from technical literature search platforms such as Google Scholar, Microsoft Academic, and IEEE Xplore. And the main content of the selected literature is relevant to the research objectives of this project, which is the study of the VPLR algorithm.

The literature analysis method is an indirect, non-interventionist survey. It only studies and examines a variety of literature, but does not contact the respondent or interfere with any of the respondent's responses. This avoids the various response errors that can occur during the interaction between investigator and respondent, which usually happen in direct surveys. Snyder (2019) illustrated that “the literature analysis method provides the basis for building a new conceptual model or theory, and it can be valuable when aiming to map the development of a particular research field over time”. Moreover, the documentary analysis method is primarily a written survey and is not limited by time or space, and if the literature collected is authentic and scholarly, it can provide more accurate and reliable information than interview methods.

**Chapter 4**

# Design, Development and Evaluation

## 4.1 Software Development Projects

### 4.1.1 Requirements Collection and Analysis

The vehicle integrated information collection system has a wide range of application scenarios, such as the vehicle's traffic management, freeway charge, traffic. therefore, the system needs to combine different application scenarios to collect the relevant functional and non-functional requirements. Generally speaking, functional [requirement](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)s such as number plate recognition, vehicle type recognition, colour recognition and vehicle counting are the most common in these systems.

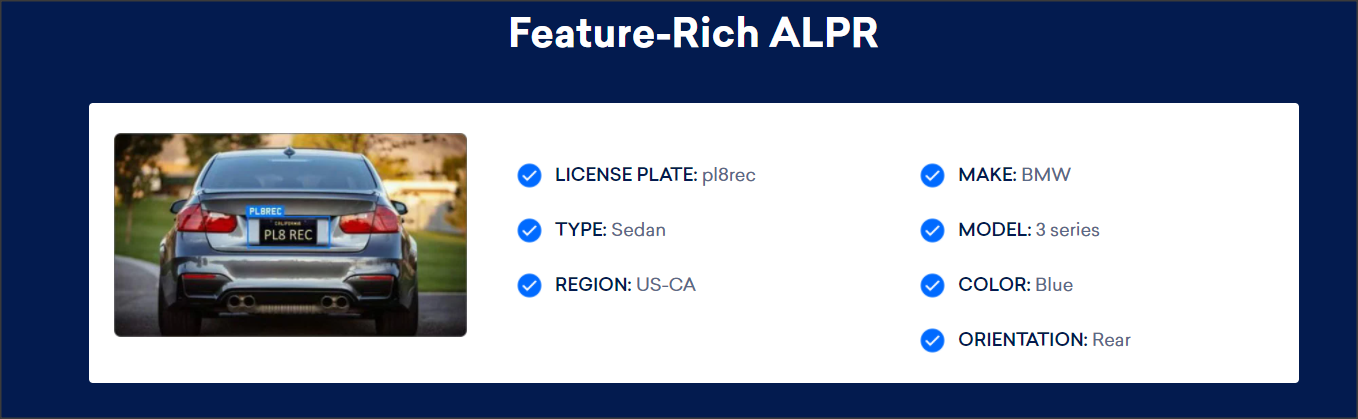


Fig. VLPR Features from Plate Recognizer

As can be seen from Fig., these functional requirements can be obtained intuitively through the analysis of some commercial VLPR or ALPR projects. There are seven functions in this VLPR system, including license plate, make, type, region, model, colour, and orientation. However, this example is a commercial project, whereas the current project is a research project, so the functionality of the product of this project is different. [For](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [instance](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;), the system is intended for using in China, so the region identification function is redundant and the orientation function is not practical for the system. So these two functions should be removed.

**Functional Requirements**

Table ...

|  |  |
| --- | --- |
| **Functions** | **Description** |
| Select Image | Select a image of the vehicle and display it in the system. |
| Plate  Positioning | Before the plate recognition, the area of the plate needs to be located in the vehicle image and extracted for the next step in the process. |
| Plate  Recognition | Individual processing of the located plate area to identify the license plate number. |
| Slant  Correction | Due to the skewed position of some vehicle plate images, it is impossible to identify the number plate accurately. Therefore, it is very remarkable that the plate correction function contributes to the accuracy of the system. |
| Make  Recognition | Identify the make of a car by its image. Such as Volkswagen, Audi, Chevrolet... |
| Model  Recognition | Identify the specific series and model of the car brand and the approximate time to market. |
| Colour  Recognition | Identifying the body colour of cars. Such as white, dark, red, blue. However, if the bodywork has multiple colours, it can be difficult to distinguish and identify |
| Vehicle  Counting | Identify the vehicles and mark them with a rectangular frame, and counting the number of vehicles passing in a video of traffic. |
| Multiple Vehicles Recognition | Accurate positioning and identification of multiple vehicles when they are shown in an image. |

**[Non-functional](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [Requirements](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)**

|  |  |
| --- | --- |
| **Functions** | **Description** |
| Usability  Requirement | The system should to be designed with an aesthetically pleasing User Interface (UI), and with human-computer interaction, operational logic. |
| Availability Requirements | The system should be available at all times and not be unavailable because of lack of internet or other reasons |
| Reliability  Requirement | The VLPR system should have an accuracy of over 90% in recognize license plates and other information about the vehicle. |
| Maintainability  Requirement | The functionality of the system can be upgraded or maintained for different usage scenarios. |
| Data Safety Requirement | In the system, the images of the cars used for identification and the final identification results should be stored securely and not leaked to the public. |

Table ...

**Development Environment**

|  |  |  |
| --- | --- | --- |
| **Hardware [Requirements](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)** | | |
| **Device** | **Basic Configuration** | **Recommended Configuration** |
| CPU | Inter: i3 - 9320  AMD: R5 - 2500X | Inter: i7 - 9700  AMD: R7 - 2700 |
| GPU | 4 GB | 8 GB |
| CPU Frequency | 2.5 ~ 2.9 GHz | 3.5 GHz |
| RAM | 8 GB | 16 GB |
| Disk Volume | 128 GB | 512GB |

Table ...

The functional requirements of the vehicle information collection system are analyzed in Table., and the non-functional requirements and hardware requirements of the project are discussed in Table. and Table. respectively in two tables. The programming stage will be developed in accordance with the collected functional and non-functional requirements

### 4.1.2 Design and Programming

**User Interface Diagram**

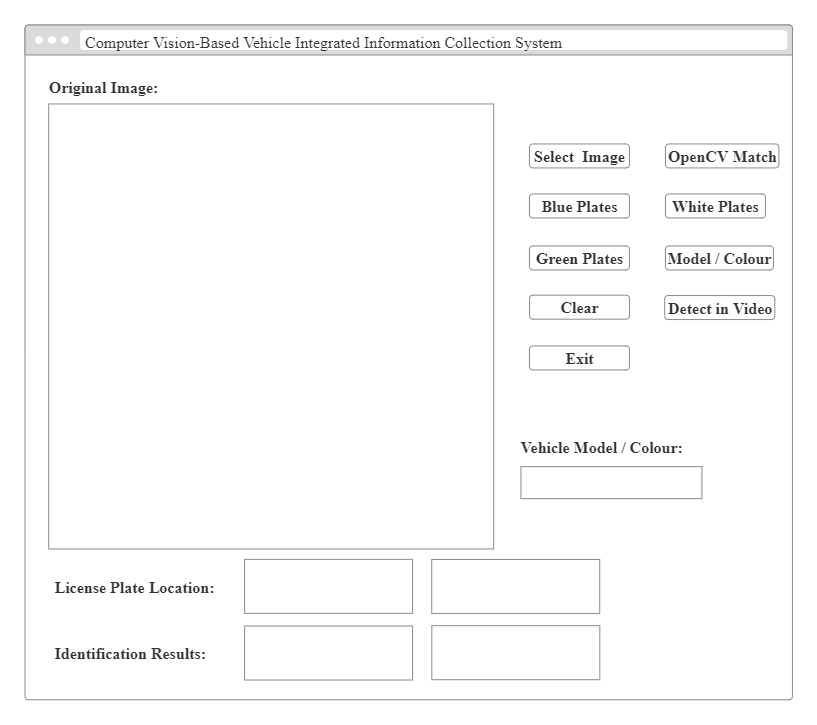


Fig. User interface diagram of this system

Fig. shows the User Interface diagram of this project product. From the UI diagram, it is clearly seen the functional design, interaction design and interface design of this product. The interface is divided into three main areas, which are the vehicle image display area, the Button control operation area and the results display area.

**[Use](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [Case](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [Diagram](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)**

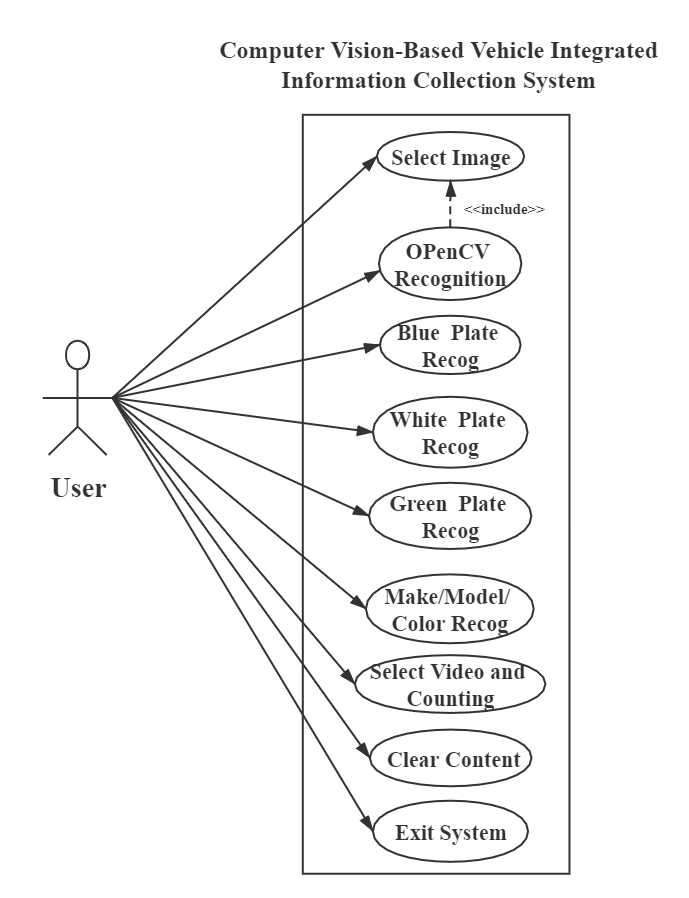


Fig. Use case diagram of this system

|  |  |  |  |
| --- | --- | --- | --- |
| **Uses Cases List** | | | |
| No. 1 | Select Image | No. 6 | Make/Model/Colour Recognition |
| No. 2 | OpenCV Recognition | No. 7 | Select Video and Counting |
| No. 3 | Blue Plate Recognition | No. 8 | Clear Content |
| No. 4 | White Plate Recognition | No. 9 | Exit System |
| No. 5 | Green Plate Recognition |  |  |

Table: Use cases list of vehicle integrated information collection system

The use case diagram is a minimalist representation of the user's interaction with the system, showing the relationship between the user and other related use cases. The use case diagram provides information about the different types of users and uses cases of the system.The use case diagram in Fig. shows that the system has nine use cases, each of which is described and analysed in detail in the tables below.

**Actor and Use Cases Details**

|  |  |
| --- | --- |
| **Use Case Name:** | Select Image |
| **Actor:** | User |
| **Description:** | The user has to open a folder on the computer and select an image of the vehicle. This photo is displayed in the system for subsequent operations. |
| **Normal Course:** | 1. User clicks the ‘Select Image’ button. 2. The system prompts the user to open the folder 3. Select a .jpg or .png image in the folder, and an error is reported if a file of a different format is selected. 4. The selected photo is displayed on the system. |
| **Pre-Condition:** | None. |
| **Post-Condition:** | The administrator has successfully selected an image and displayed it. |

|  |  |
| --- | --- |
| **Use Case Name:** | OpenCV Recognition |
| **Actor:** | User |
| **Description:** | License plate recognition using the OpenCV approach. A photo of the vehicle needs to be selected before recognition. |
| **Normal Course:** | 1. User clicks the ‘OpenCV Match’ button. 2. The system prompts the user to open the folder 3. Select a .jpg or .png image in the folder, and an error is reported if a file of a different format is selected. 4. After the image processing and template matching operations, the final recognition results are displayed in the system. |
| **Pre-Condition:** | None. |
| **Post-Condition:** | The selected car image is displayed in the system, showing the recognition result (success or failure). |

|  |  |
| --- | --- |
| **Use Case Name:** | Blue Plate Recognition |
| **Actor:** | User |
| **Description:** | License plate location and plate recognition for vehicles with blue plates using another method |
| **Normal Course:** | 1. Vehicle picture is successfully displayed in the system. 2. User clicks the ‘Blue Plate’ button. |
| **Pre-Condition:** | Select Image, Clear Content |
| **Post-Condition:** | Displaying a image of the plate area and the identified plate number in the system |

|  |  |
| --- | --- |
| **Use Case Name:** | White Plate Recognition |
| **Actor:** | User |
| **Description:** | License plate location and plate recognition for vehicles with white plates using another method |
| **Normal Course:** | 1. Vehicle picture is successfully displayed in the system. 2. User clicks the ‘White Plate’ button. |
| **Pre-Condition:** | Select Image, Clear Content |
| **Post-Condition:** | Displaying a image of the plate area and the identified plate number in the system. |

|  |  |
| --- | --- |
| **Use Case Name:** | Green Plate Recognition |
| **Actor:** | User |
| **Description:** | License plate location and plate recognition for vehicles with green plates using another method. |
| **Normal Course:** | 1. Vehicle picture is successfully displayed in the system. 2. User clicks the ‘Green Plate’ button. |
| **Pre-Condition:** | Select Image, Clear Content |
| **Post-Condition:** | Displaying a image of the plate area and the identified plate number in the system. |

|  |  |
| --- | --- |
| **Use Case Name:** | Make/Model/Colour |
| **Actor:** | User |
| **Description:** | Identify the make, model, colour and release date of the car in the image and display the results. |
| **Normal Course:** | 1. Vehicle picture is successfully displayed in the system. 2. User clicks the ‘Model / Colour’ button. |
| **Pre-Condition:** | Select Image, Blue Plate Recognition, White Plate Recognition, Green Plate Recognition. |
| **Post-Condition:** | The make, series, model, colour and release date of the vehicle are displayed in detail. |

|  |  |
| --- | --- |
| **Use Case Name:** | Select Video and Counting |
| **Actor:** | User |
| **Description:** | Detects passing vehicles in the video and marks and counts them. |
| **Normal Course:** | 1. User clicks the ‘Detect in Video’ button. 2. The system prompts the user to open the folder 3. Select a .mp4 video in the folder, and an error is reported if a file of a different format is selected. 4. Play this video |
| **Pre-Condition:** | None. |
| **Post-Condition:** | The selected video is processed and the vehicles in the video are detected and marked with a frame. The total number of cars passing by is then calculated. |

|  |  |
| --- | --- |
| **Use Case Name:** | Clear |
| **Actor:** | User |
| **Description:** | Clear images and data information from the system interface. |
| **Normal Course:** | 1. User clicks the ‘Clear’ button. |
| **Pre-Condition:** | Select Image, OpenCV Recognition, Blue Plate Recognition, White Plate Recognition, Green Plate Recognition, Make / Model / Colour Recognition. |
| **Post-Condition:** | The vehicle's picture, plate number, make, model and colour of the vehicle are all cleared |

|  |  |
| --- | --- |
| **Use Case Name:** | Exit System |
| **Actor:** | User |
| **Description:** | Exit the system |
| **Normal Course:** | 1. User clicks the ‘Exit’ button. 2. Pop-up prompt: Confirm to exit the program? |
| **Pre-Condition:** | None. |
| **Post-Condition:** | Exit from the system successful or cancel this operation. |

**Activity Diagrams**

Another important behavioral diagram in the UML diagram is the activity diagram, which is used to illustrate complex features of the system. It illustrates the workflow for business use case implementation. And The activity diagram for each use case is shown from Fig. to Fig.

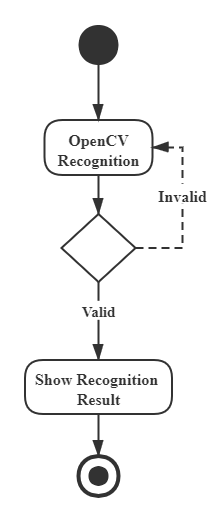
 

Fig. : Make/Model/Colour Recognition’ and ‘Select Image’ Activity Diagrams

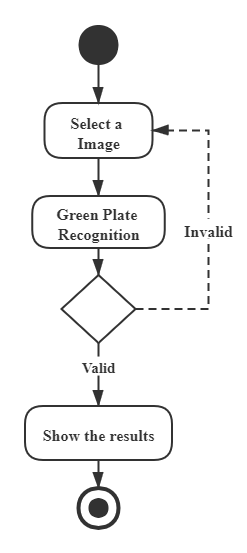
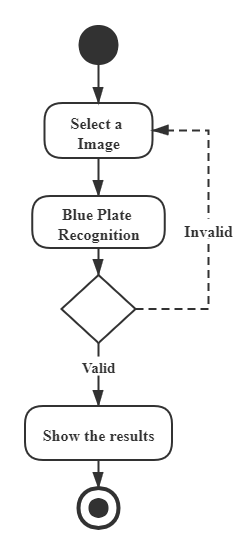
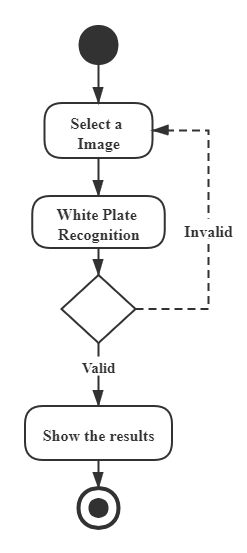
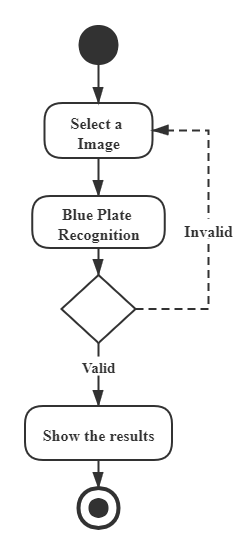
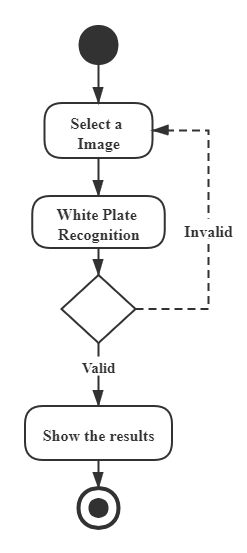


Fig. : ‘White Plate Recognition’, ‘Blue Plate Recognition’ and ‘Green Plate Recognition’ Activity Diagrams



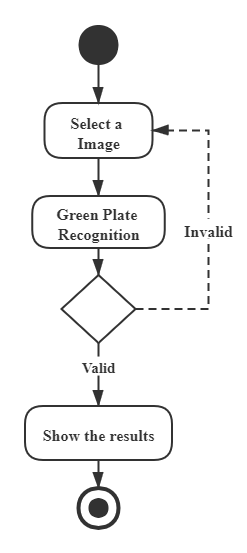
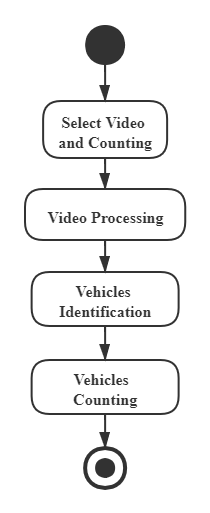
 

Fig. : ‘Make/Model/Colour Recognition’, ’ and ‘Select Video and Counting’ Activity Diagrams

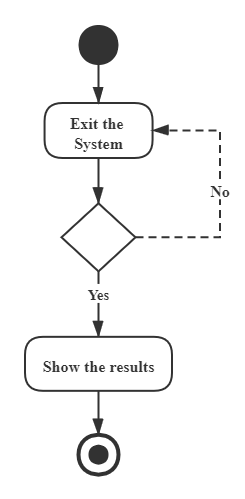
 

Fig. : ‘Clear Content’, ’ and ‘Exit System’ Activity Diagrams

**Sequence Diagrams**

The sequence diagram shows the dynamic collaboration between multiple objects by depicting the temporal order in which messages are sent between them. It can represent the order of behaviour of a use case. The sequence diagrams Fig. to Fig. show the interaction between different objects and use cases.

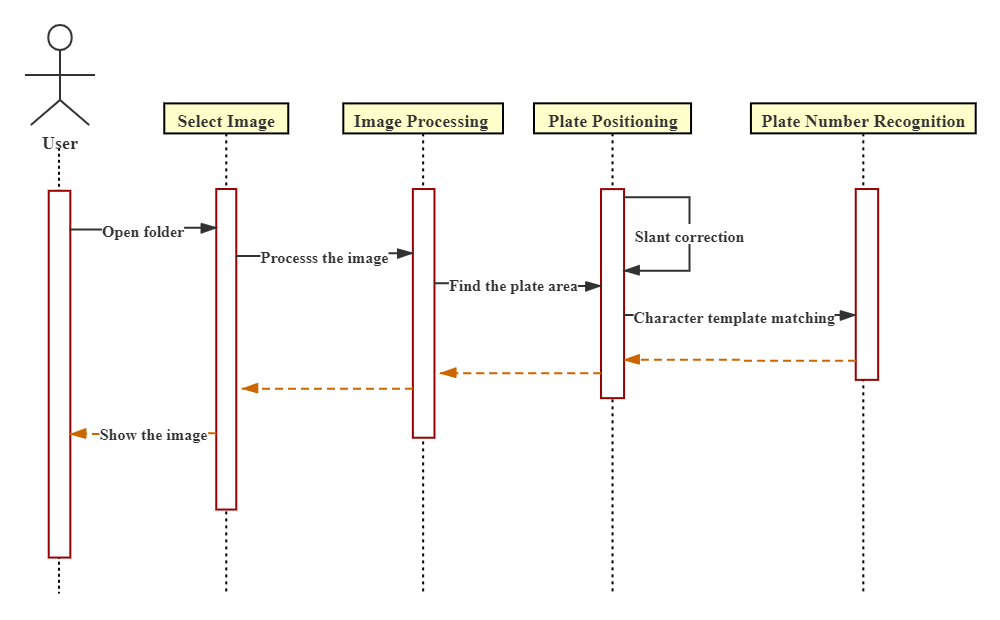
****

Fig. Sequence diagram of Open CV Recognition use case

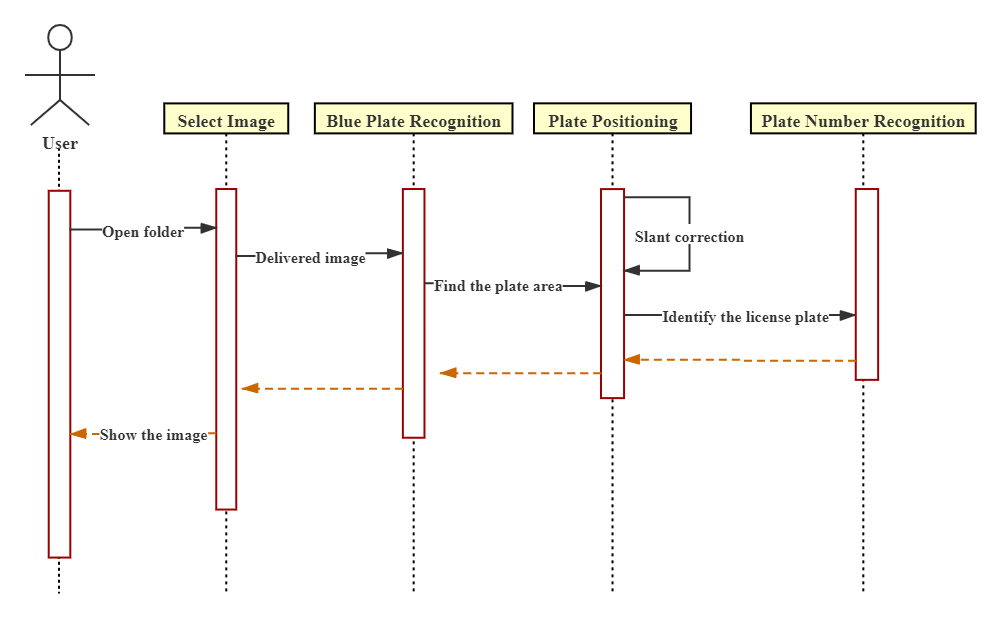
****

Fig. Sequence diagram of Blue Plate Recognition use case

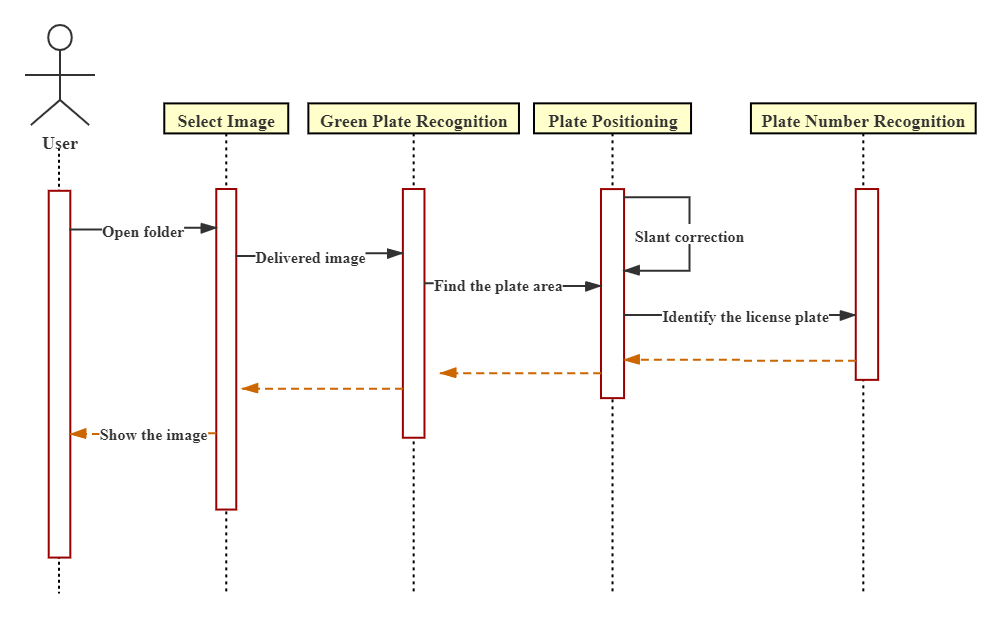
****

Fig. Sequence diagram of Green Plate Recognition use case

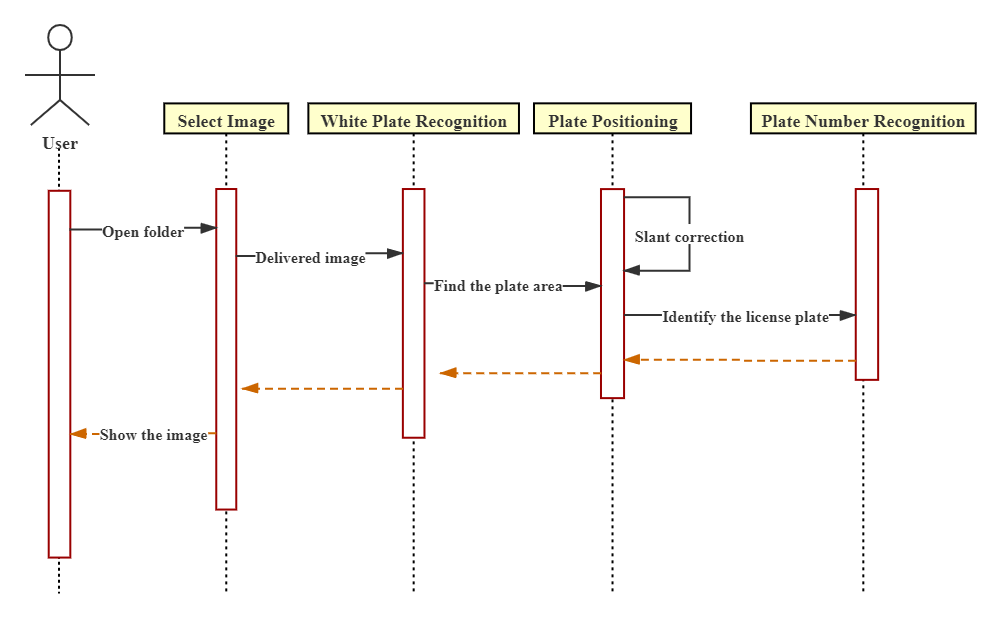
****

Fig. Sequence diagram of White Plate Recognition use case

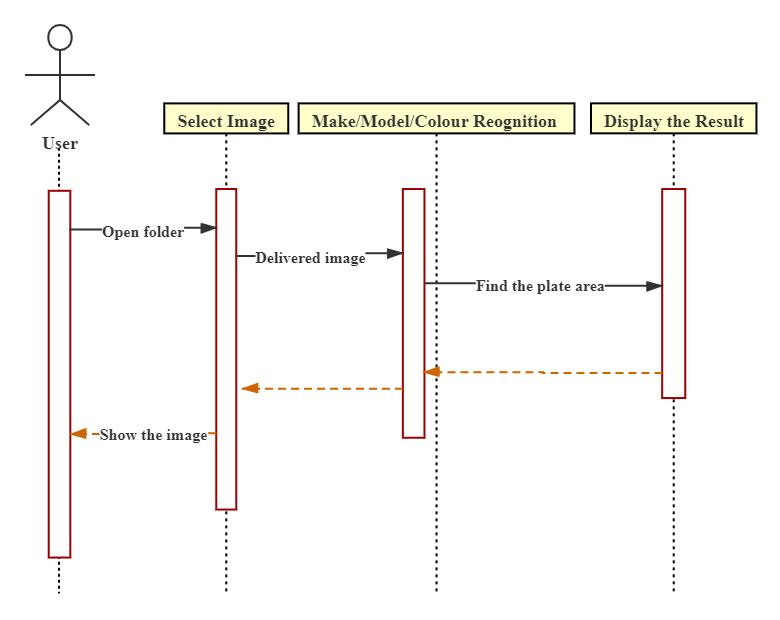
****

Fig. Sequence diagram of Make/Model/Colour Recognition use case

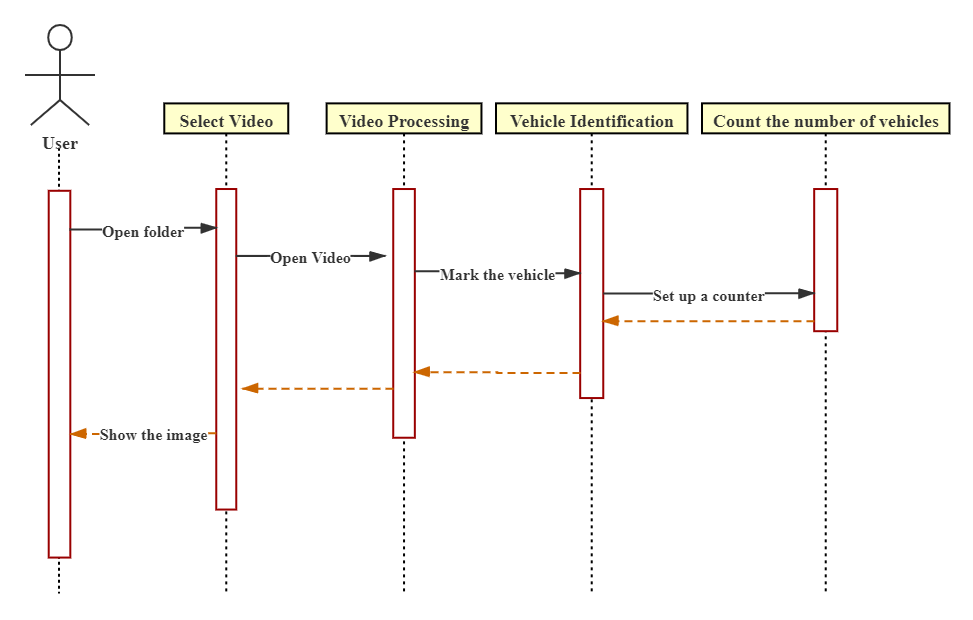
****

Fig. Sequence diagram of Select Video and Counting use case

**Functions [and Algorithm](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) Design**

|  |  |
| --- | --- |
| Class | Select Images |
| Method | load\_show\_img(self) |
| Purpose | Open the folder and load a vehicle image into this system |
| Per Condition | N / A |
| Post Condition | The image is shown on the system |
| Return Value | N / A |
| Coupling |  |
| Input | File path of images |
| Output | images |
| Storage | N / A |
| Pseudo-code:  Image.path = Entry() // Get the images path on the computer  Image.open (Image.path)  If image.size != the requirements  Image.resize(length, width)  Image.show() | |

|  |  |
| --- | --- |
| Class | Clear |
| Method | clear(self) |
| Purpose | Delete the image, data or information |
| Per Condition | There are some data displayed on the system |
| Post Condition | All the data have been deleted. |
| Return Value | N / A |
| Coupling |  |
| Input | N / A |
| Output | N / A |
| Storage |  |
| Pseudo-code:  Control.text.delete(‘all’) | |

|  |  |
| --- | --- |
| Class | OpenCV\_functions |
| Method | pre\_processing() |
| Purpose | Pre-process vehicle image to get plate positioning before recognition |
| Per Condition | Select Image |
| Post Condition | Image processing |
| Return Value | N / A |
| Coupling |  |
| Input | Vehicle image |
| Output | The license plate image |
| Storage | N / A |
| Pseudo-code:  Image = Vehicle image.copy()  Image = Image.gauss noise processing()  Image = Image.gray processing()  Image = [Image](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [enhancement](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)()  Image = [Morphological](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [processing](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)()  Image = Median blur noise reduce()  Image = [Edge](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [detection](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)()  OpenCV.draw Contours()  If (contours.width> 3 \* length) and (width < 4\*length)  Image = image.cv2.bounding Rect  Show the plate image | |

|  |  |
| --- | --- |
| Class | OpenCV\_functions |
| Method | license\_spilt(image); template\_matching |
| Purpose | Split the plate image and recognize the characters and numbers |
| Per Condition | Image processing and plate positioning |
| Post Condition | Image processing |
| Return Value | results |
| Coupling |  |
| Input | The license plate image |
| Output | [license](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [plate](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [number](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) |
| Storage | N / A |
| Pseudo-code:  Plate image.pre-processing()  Plate\_image.histogram.show()  Histogram.split()  Get the word image of each individual character  Return the word\_images  For i in times.plate.numbers:  For word\_images in template library  Get similar score of each images  Best\_score = the similar.score.max()  Word = best\_score.index()  Result = word.append()  Return result | |

|  |  |
| --- | --- |
| Class | Unet |
| Method | unet.train(); unet\_predict(unet, img\_src\_path) |
| Purpose | Set an U-net model and train it to achieve license plate positioning |
| Per Condition | N / A |
| Post Condition | Get a trained U-net model, and predict the location of license plate |
| Return Value | N / A |
| Coupling |  |
| Input | Training images dataset, Vehicle image |
| Output | U-net Model, license plate image |
| Storage | N / A |
| Pseudo-code:  Create an U-net Model  [Configuration](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [parameter](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)()  Model.optimizer()  Model.compile()  Model.fit(train\_input, train\_output, epochs.value, batch\_size)  U\_net = Model.save(File Path) // Get a U-net model  Image = U\_net.predict(vehicle\_image)  Get the image of license plate | |

|  |  |
| --- | --- |
| Class | CNN |
| Method | cnn\_train(); cnn\_predict() |
| Purpose | Create a CNN model and train it to achieve [license](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [plate](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [recognition](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;). |
| Per Condition | U-net model has been set up and trained |
| Post Condition | Get a trained CNN model, and predict license plate number |
| Return Value | license\_plate\_number |
| Coupling |  |
| Input | Training images dataset, Vehicle plate image |
| Output | CNN Model; license plate number |
| Storage | N / A |
| Pseudo-code:  Create an CNN Model  [Configuration](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [parameter](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)()  Model.optimizer()  Model.compile()  Model.fit(train\_input, train\_output, epochs.value, batch\_size)  CNN = Model.save(File Path) // Get a CNN model  Image\_predict = CNN.predict(license\_Plate\_image)  if image\_predict probability value > 80%  Get the predicted number  Return license\_plate\_number | |

|  |  |
| --- | --- |
| Class | image\_slant\_correction |
| Method | license\_image(); boundingRect\_draw(); boundingRect\_show()  calculate\_slope(); image\_rotation() |
| Purpose | Automatic correction of skewed vehicle photos |
| Per Condition | N / A |
| Post Condition | Display of a corrected vehicle number plate |
| Return Value | license\_plate |
| Coupling |  |
| Input | Vehicle image |
| Output | Corrected vehicle number plate |
| Storage | N / A |
| Pseudo-code:  Vehicle\_image = imread(image.path)  Plate\_contour = bounding\_Rect(Vehicle\_image)  K = Calculate\_slop of license plate  Image = rotation(K) | |

|  |  |
| --- | --- |
| Class | VehicleTracker |
| Method | ObjectTracker() |
| Purpose | Play a video and detect passing vehicles in the video and marks and counts them. |
| Per Condition | N / A |
| Post Condition | The selected video is processed and the vehicles in the video are detected and marked with a frame. The total number of cars passing by is then calculated. |
| Return Value | N / A |
| Coupling |  |
| Input | Vehicle Video |
| Output | The number of vehicles |
| Storage | N / A |
| Pseudo-code:  ObjectID = 0  [modeling](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;).BackgroundSubtractorMOG2(Video)  for i in [frame](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [number](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;):  median filtering()  Open Operation() // [morphological](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [processing](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;)  Close Operation()  Get the contour  Rect\_Framework.draw()  ObjectID += 1  Read next frame | |

**Building and Programming**

Suitable techniques and methods will be used during the programming stage to implement the collected requirements and the designed product. Also, iterative incremental development methods will be used, with 3 times iterations to achieve the final complete product. From Table. to Table. the features that should be developed in each iteration are listed. The functional structure diagrams of each iteration are shown in Fig. , Fig. , and Fig. , respectively.

|  |  |
| --- | --- |
| **The 1st Iteration Development** | |
| **Task** | **Description** |
| Select Images | Open the folder and select a vehicle image to display |
| OpenCV Recognition | Traditional license plate recognition using technologies such as OpenCV vision libraries and OCR |
| U-net Model for blue license plate | Training a U-net neural network model for blue license plate location and segmentation |
| CNN Model for blue license plate | Training a CNN model for blue license plate recognition. |
| Clear Content | Delete all the data on the system |
| Exit System | Exit the system |

### Table. The first iteration development at programming step

|  |  |
| --- | --- |
| **The 2nd Iteration Development** | |
| **Task** | **Description** |
| Make/Model/Colour/  Year Recognition | Identify the shown vehicle photo and output the vehicle's make, model, colour and year |
| U-net Model for white license plate | Training a U-net neural network model for white license plate location and segmentation |
| CNN Model for white license plate | Training a CNN model for white license plate recognition. |
| Image Slant Correction | Slant correction for segmented plate photos |

### Table. The second iteration development at programming step

|  |  |
| --- | --- |
| **The 3rd Iteration Development** | |
| **Task** | **Description** |
| Detect vehicles and counting | Detects passing vehicles in the video, marking and counting them. |
| U-net Model for green license plate | Training a U-net neural network model for green license plate location and segmentation |
| CNN Model for green license plate | Training a CNN model for green license plate recognition. |

### Table. The third iteration development at programming step

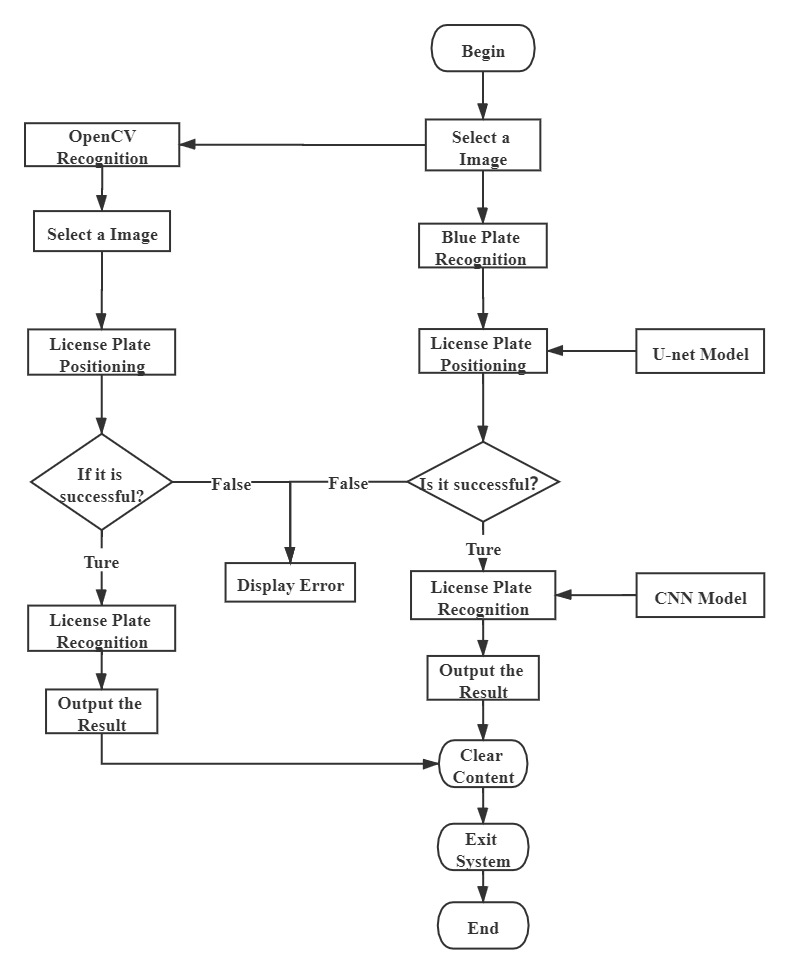


Fig. The function structure diagram of first iteration development



Fig. The function structure diagram of second iteration development

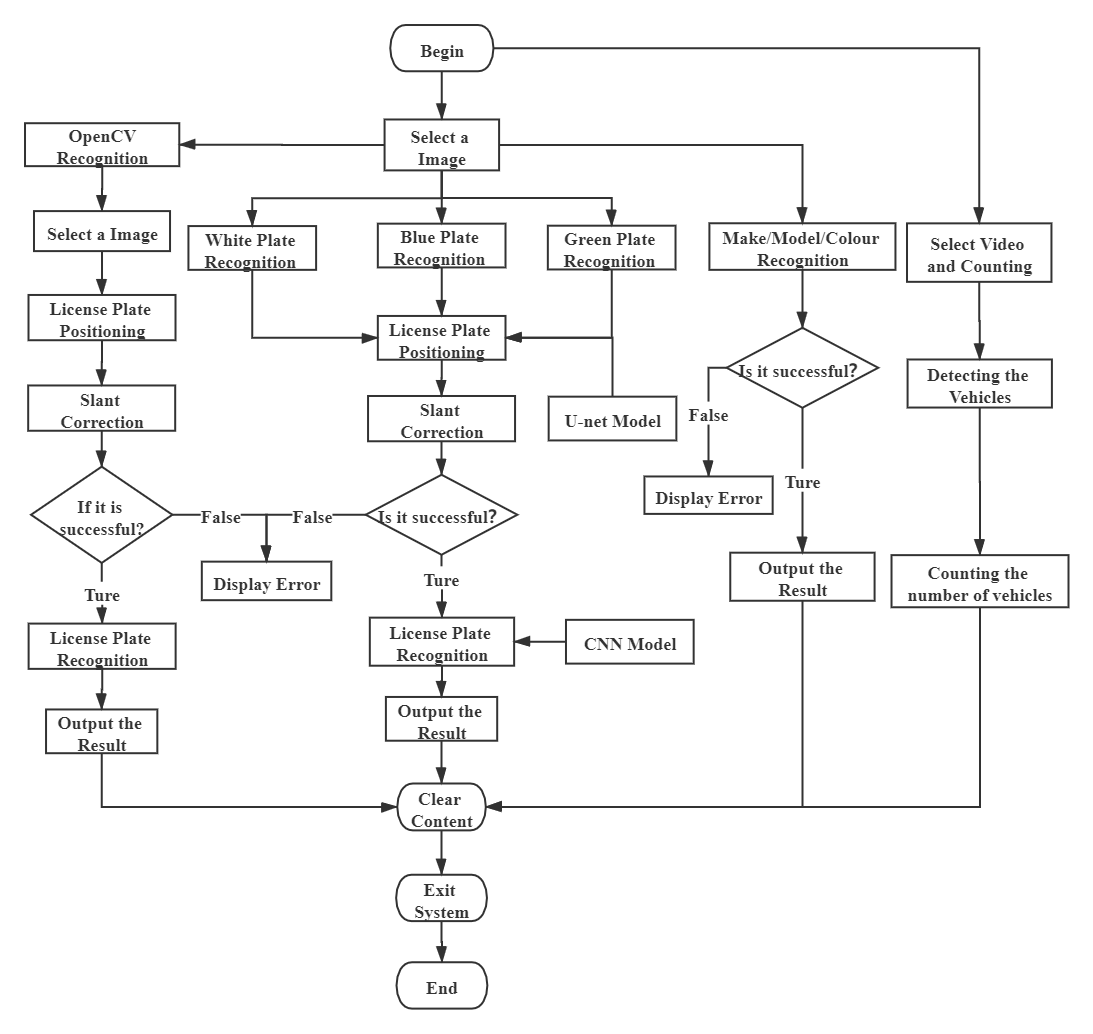


Fig. The function structure diagram of third iteration development

### 4.1.3 Testing

The purpose of this test is to complete the testing of the entire system, verifying basic usability, functional integrity, data accuracy and maintainability of the software. There are a number of testing methods that can be used, such as white-box testing, black-box testing, static testing, dynamic testing, unit testing, and integration testing. In the testing section, the use cases from the three iterations of development are combined into a detailed test plan and each use case is tested in detail.

For this system, black box testing is the primary testing method, which is the most frequently used testing method in which testers evaluate the functionality of the system under test without looking at the internal code structure. Because black-box testing is suitable for large code segments, and the testers can be non-technical, this testing technique does not require knowledge of programming. However, it also has some limitations, Ehmer and Khan (2012) indicated that “only a selected number of test scenarios are actually performed. As a result, there is only limited coverage”. Therefore, in order to improve the quality of software, a combination of testing methods should be used to find bugs and errors more efficiently.

**Testing Plan**

Test Member: Shangyuan Liu Test Date: 24 / 04 / 2021

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Controller** | **Module** | **Function** |
| 1 | User | Select Images | Open a .png format image and show |
| 2 | Open a .jpg format image |
| 3 | Open the other format file |
| 4 | OpenCV  Match | Select a full-face image of vehicle |
| 5 | Select a back of vehicle image |
| 6 | Select a slant vehicle image |
| 7 | Blue Plate Recognition | Select a full-face image of vehicle |
| 8 | Select a back of vehicle image |
| 9 | Select a slant vehicle image |
| 10 | Select a license plate image |
| 11 | Select a image with two vehicles |
| 12 | Green Plate Recognition | Select a full-face image of vehicle to segment plate |
| 13 | Select a back of vehicle image to segment plate |
| 14 | Select a slant vehicle image to segment plate |
| 15 | Select a license plate image to segment plate |
| 16 | Select a image with two vehicles to segment plate |
| 17 | White Plate Recognition | Select a full-face image of vehicle to segment plate |
| 18 | Select a back of vehicle image to segment plate |
| 19 | Select a slant vehicle image to segment plate |
| 20 | Select a license plate image to segment plate |
| 21 | Select a image with two vehicles to segment plate |
| 22 | Model/Colour Recognition | Select a full-face image of vehicle |
| 23 | Select a back of vehicle image |
| 24 | Detect in Video | Open a .mp4 video to play and process |
| 25 | Clear Content | Clear all the data |
| 26 | Exit System | Exit the system |

**Testing Case**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | 1 | 2 | 3 |
| Input | test1.png | test2.jpg | test.py |
| Screenshot | Snap 2021-05-03 at 23.00.18 Snap 2021-05-03 at 23.07.15 Snap 2021-05-03 at 23.07.43 | | |
| Result | Pass | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | 4 | 5 | | | 6 |
| Input | full-face image | image of back of vehicle | | | slant vehicle image |
| Screenshot | **Snap 2021-05-03 at 23.19.43 Snap 2021-05-03 at 23.20.38 Snap 2021-05-03 at 23.21.11** | | | | |
| Result | Pass | | Pass | Fail | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | 7 | 8 | | | 9 |
| Input | full-face image | image of back of vehicle | | | slant vehicle image |
| Screenshot | **Snap 2021-05-03 at 23.34.14 Snap 2021-05-03 at 23.34.37 Snap 2021-05-03 at 23.34.57** | | | | |
| Result | Pass | | Pass | Pass | |

|  |  |  |
| --- | --- | --- |
| Test ID | 10 | 11 |
| Input | license plate image | A image with two vehicles |
| Screenshot | **Snap 2021-05-04 at 00.17.53 Snap 2021-05-04 at 00.11.16** | |
| Result | Pass | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | 12 | 13 | | | 14 |
| Input | full-face image | image of back of vehicle | | | slant vehicle image |
| Screenshot |  | | | | |
| Result | Pass | | Pass | Pass | |

|  |  |  |
| --- | --- | --- |
| Test ID | 15 | 16 |
| Input | license plate image | A image with two vehicles |
| Screenshot | **Snap 2021-05-04 at 01.18.38 Snap 2021-05-04 at 01.29.42** | |
| Result | Fail | Pass |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | 17 | 18 | | | 19 |
| Input | full-face image | image of back of vehicle | | | slant vehicle image |
| Screenshot |  | | | | |
| Result | Pass | | Pass | Pass | |

|  |  |  |
| --- | --- | --- |
| Test ID | 20 | 21 |
| Input | license plate image | A image with two vehicles |
| Screenshot | **Snap 2021-05-04 at 00.41.13 Snap 2021-05-04 at 00.37.42** | |
| Result | Fail | Pass |

|  |  |  |
| --- | --- | --- |
| Test ID | 22 | 23 |
| Input | full-face image | image of back of vehicle |
| Screenshot |  | |
| Result | Pass | Pass |

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | 24 | 25 | 26 |
| Input | .mp4 video | | |
| Screenshot | **Snap 2021-05-04 at 02.03.54** | | |
| Result | Pass | | |

### 4.1.4 Operation

**Program Flow Diagram**

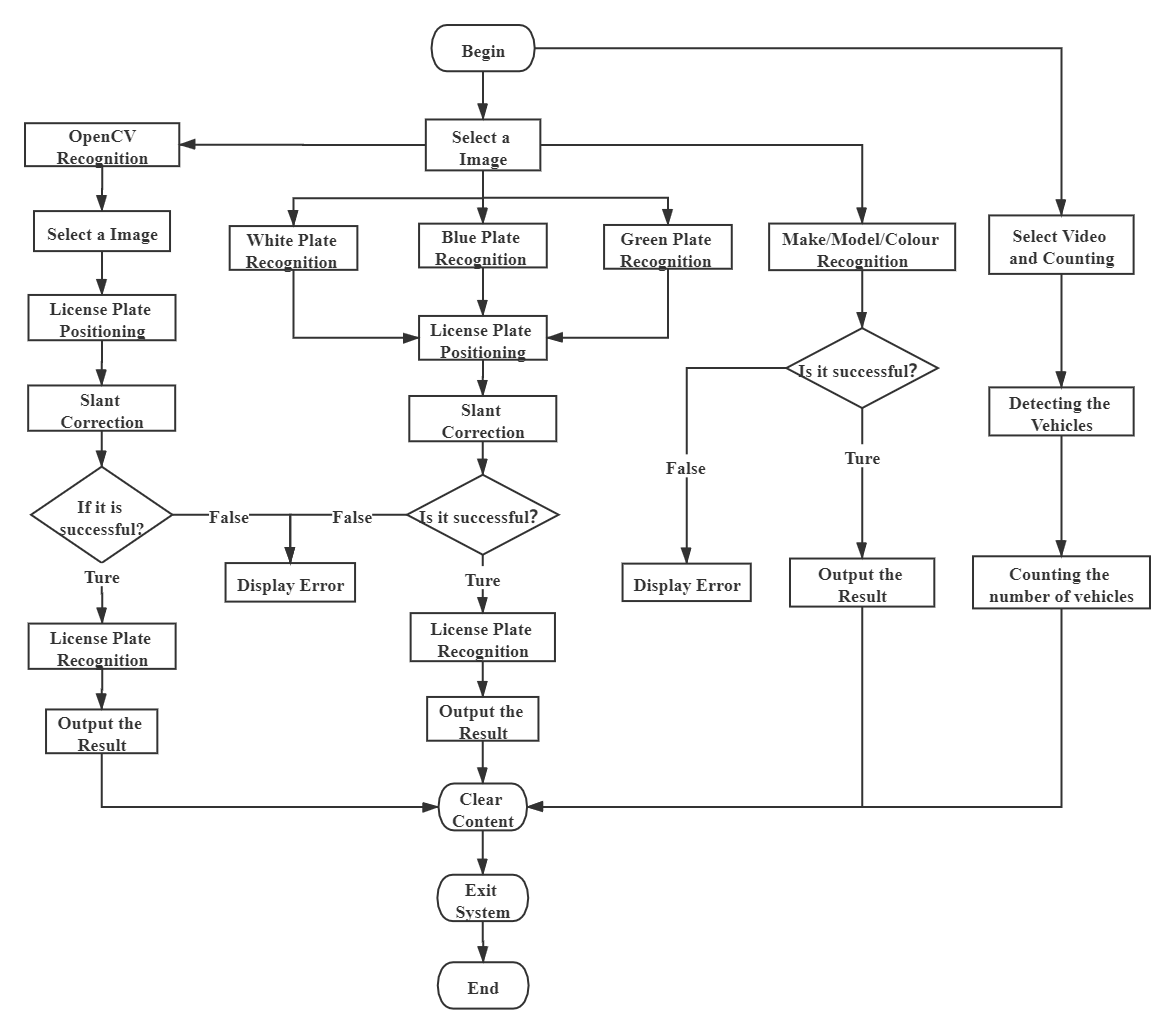


Fig. Use case diagram of this system

The program flow diagram illustrates the data flow of the system or algorithm and the flow diagram also analyses the logic behind the program. It can be seen from the Fig. that flow relationships between the different functions in the system.

## 4.2 Research Projects

### 4.2.1 Methods

**Research of VLPR based on OpenCV**

The traditional VLPR technology is based on OpenCV, which includes image processing and Optical Character Recognition. The algorithm flowchart for this method is shown in Fig., using OpenCV to pre-process the image, and filter the contours of the licence plate position, then perform character segmentation, match each character image with the character template library, and finally output the recognition result. However, this technique is gradually being replaced due to its slow recognition speed, low accuracy.

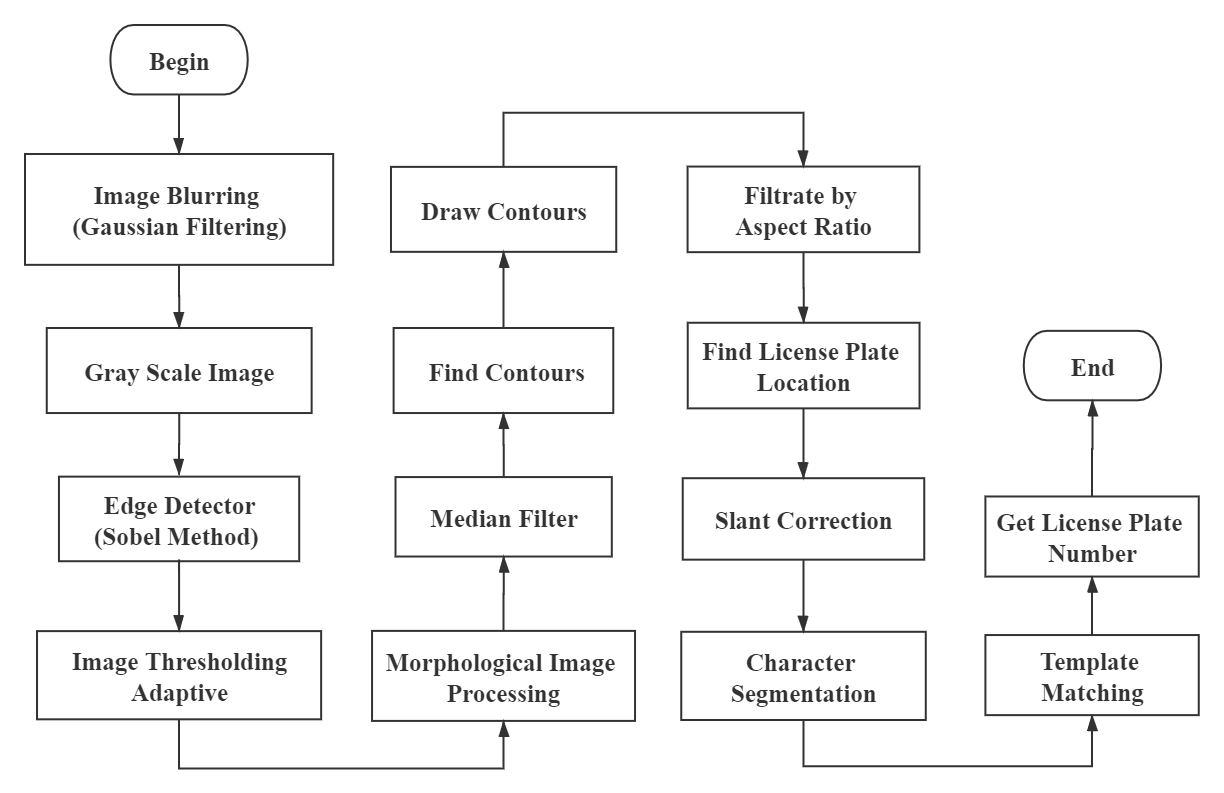


Fig. The [algorithm](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) flow diagram of OpenCV method

**Image Pre-processing:** Using Gaussian blur (1) to reduce image noise. After Gaussian noise reduction and grayscale processing, the Sobel operator (2) method is used for edge detection in the vertical (Y) direction. Where A is the original source image, the x coordinate is defined as the increase in the right direction and the y coordinate is defined as the increase in the down-direction. The OTSU algorithm is used to perform automatic image thresholding, and the binarization process enhances sharp edges and further removes useless information from the image. All the steps can be seen from Fig.

(1)

& (2)

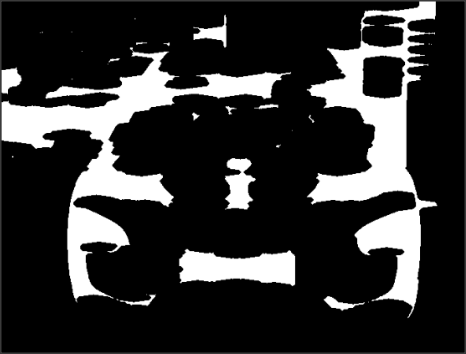
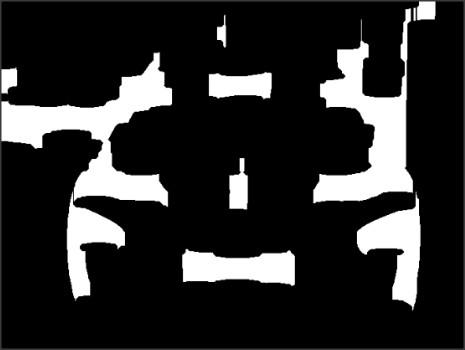
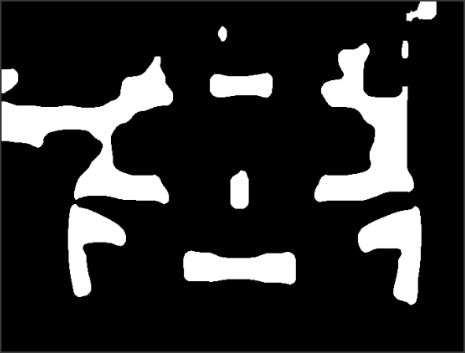


Fig. Image processing steps

**Mathematical Morphology:** In order to locate the outline of the plate more accurately, mathematical morphology was performed on the binary image, linking similar areas of the image by expansion (5) and removing small isolated blocks of colour by erosion (4). The purpose of the closing operation is to connect the objective area as a unit, and after several erosion and expansion operations, the contour lines become smoother for subsequent contour extraction.

(3)

(4)

**[Contour](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [Detection](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;):** Median Filtering is used to eliminate the image noise again, the findContours() method in OpenCV is used to detect the contours of the object, and drawContours() is used to draw the contours on the original image. In addition, the contours are traversed in order to filter out the location of the licence plate, and when the aspect ratio of the contour is 3:1 or 4:1, then this contour is the location of the licence plate.

Fig. License plate contour

**Slant Correction:** However, some vehicle images are skewed and do not enable direct recognition operations, so the skewed images have to be corrected. The fitLine() function (5) in OpenCV is used to complete the straight line fitting algorithm. Parameters such as contour coordinate points and Euclidean distances (6) are input to the function and calculate the direction vector (vx, vy) and the points (x, y) on the line. These can be used to calculate the slope and Y-intercept (7) of line. Using these parameters, the line can be drawn in the diagram (Fig.). Finally, the function cv.getRotationMatrix2D() can be used to rotate the vehicle image.

(5)

(6)

(7)

Fig. Slant plate detect and straight-line fitting



It is clear from Fig. that the corrected image is processed again to obtain a corrected plate image for subsequent recognition.

Fig. After corrected license plate

**[Character](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) [Segmentation](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;):** The plate image is pre-processed to provide a binary image with a black background and white characters. Calculating the number of black and white points in the binary image, using the closure operation to make the white part into a whole. Then filter the contours of each character's position, and finally use the cv2.boundingRect() function to draw a rectangle around the character, and appending each rectangles together as shown in Fig.

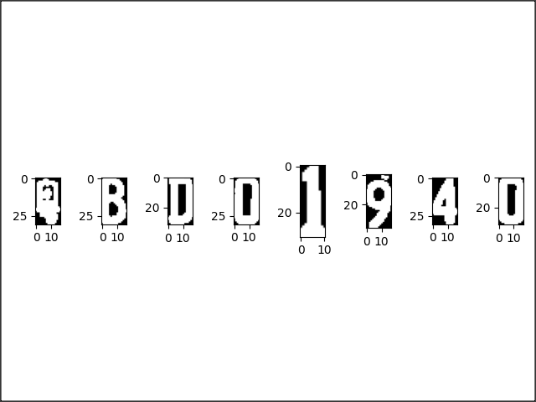


Fig. Characters segmentation result

**Optical Character Recognition:** Comparing the processed individual character images with the template image dataset, using the cv.matchTemplate() function will calculate the correlation coefficient (8) between the images, the higher the correlation coefficient the more similar it is. Therefore, the maximum correlation coefficient and its subscript are calculated, and the result is the value of the corresponding subscript position of the template that is shown in Fig.

(8)

Snap 2021-05-05 at 01.07.40

Fig. Final recognition result of the license plate

The traditional detection algorithms are complicated calculations, poorly ability of anti-jamming and do not have good real-time performance, robustness and accuracy, and the recognition failed scenario is shown in Fig. Therefore, this method does not meet the daily requirements in special environments, “those techniques were gradually replaced by powerful deep learning modules such as convolutional neural networks.” (Abdussalam et al., 2018).

Fig. Examples of failed recognition

**Research of VLPR based on computer vision (Deep Learning)**

This VLPR system is designed for Chinese license plate character recognition. Because the license plates contain Chinese characters in addition to common alphanumeric characters, robust and efficient deep learning models are essential for performing the number plate location and character recognition tasks effectively. As can be seen from the algorithm flow diagram Fig. this solution uses two deep learning models, U-Net and CNN, which are used for license plate detection and license plate recognition respectively.



Fig. The [algorithm](D:/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) flow diagram of Deep Learning method

During the training process of an end-to-end deep learning model, a prediction is obtained from the input end to the output end, and an error is received when compared to the real result. This error is passed through each layer of the model (back propagation) and the representation of each layer is adjusted according to this error until the model converges or achieves the expected result. And all the operations in between are contained within the neural network is no longer split into multiple modules for processing. From the original data input to the resulting output, from the input end to the output end, the neural network in between is self-contained.

**Label Training Set:** Before training the model it is necessary to use the image annotation tool lambel to label the image dataset. After labelling the license plate region, a corresponding JSON file is generated, and it is converted into the corresponding data file, which includes the required PNG label image (Fig. ). The system is designed to detect and recognise blue, green and white licence plates in China. Therefore, in order to improve accuracy and robustness, more than one thousand marker images are prepared for each training set.

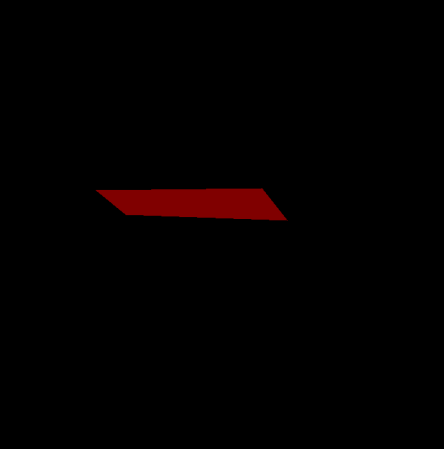
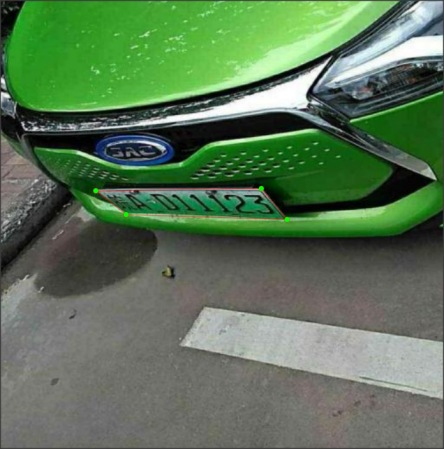
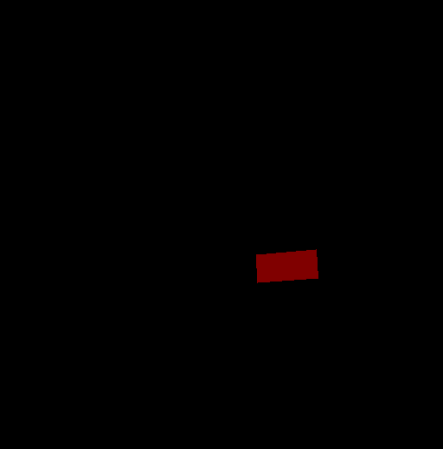
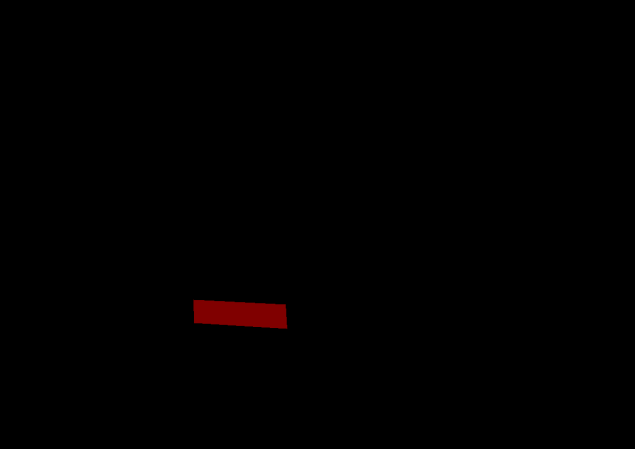


Fig. The labeled training images by labelme

**U-Net:** U-Net is a network based on full convolutional network (FCN) for image segmentation, which learns segmentation in an end-to-end setting. The structure of the U-net network is plotted in Fig. Each blue box corresponds to a multi-channel feature map, where the number of channels is shown at the top of the box and the x-y size is located in the bottom left corner of the box. The white boxes represent the duplicated feature maps and the arrows show the different operations.

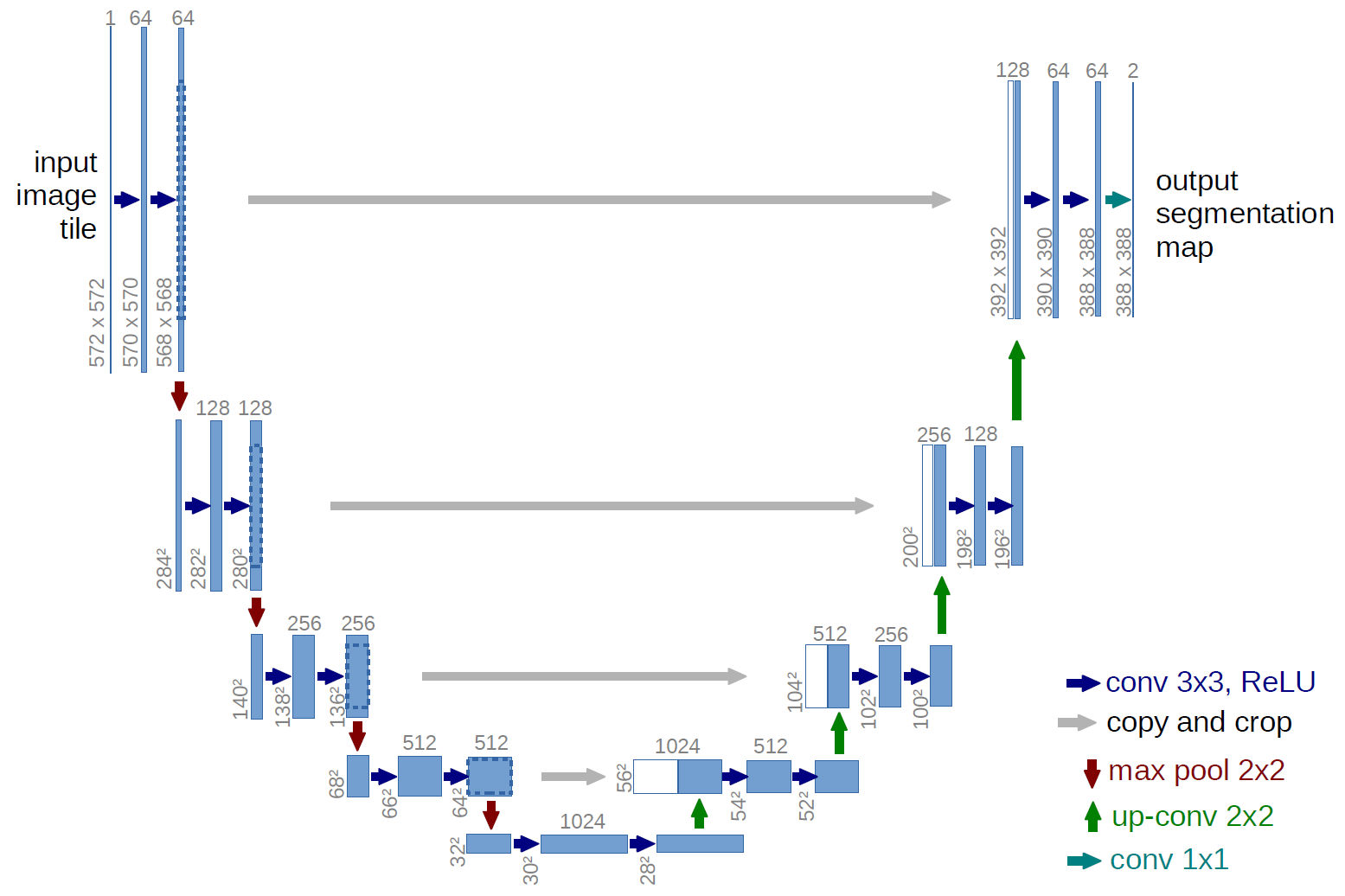


Fig. U-Net Network Structure

This network consists of a contracting path and an expanding path. The contracting path is a typical convolutional network for feature extraction and the expanding path is used for accurate localisation, and both paths are symmetrical to each other which provides a u-shaped architecture. Usually, the shrinkage network is supplemented by successive layers in which the upsampling operation replaces the pooling. As a result, these layers increase the resolution of the output results. To perform the localisation operation, high-resolution features from the shrinkage path are combined with upsampling.(Long et al., 2015). Therefore, the study by Ronneberger et al. (2015) provided this model and indicated that “a successive convolution layer can then learn to assemble a more precise output based on this information”.

U-Net is significantly different from other conventional segmentation networks in that it uses a different approach to feature fusion (concatenate), where features are stitched together in the channel dimension to form thicker features. However, U-Net runs are inefficient and each patch has to be run individually once, so it is slow. Moreover, due to the overlapping of patches, many runs are redundant and consume a lot of graphic memory. Also, “there is a trade-off between localization accuracy and the use of context. Larger patches require more max-pooling layers that reduce the localization accuracy.” (Ronneberger et al., 2015).

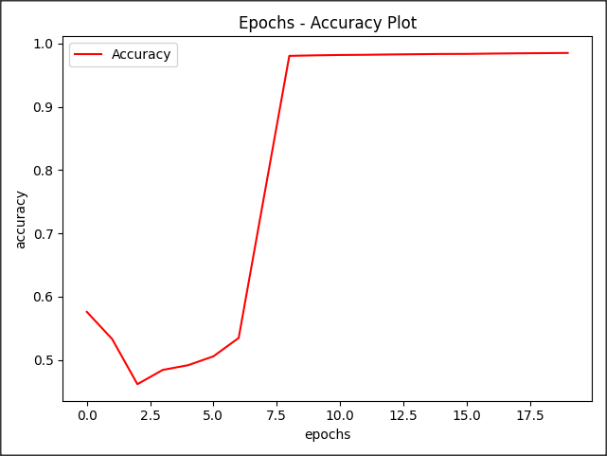
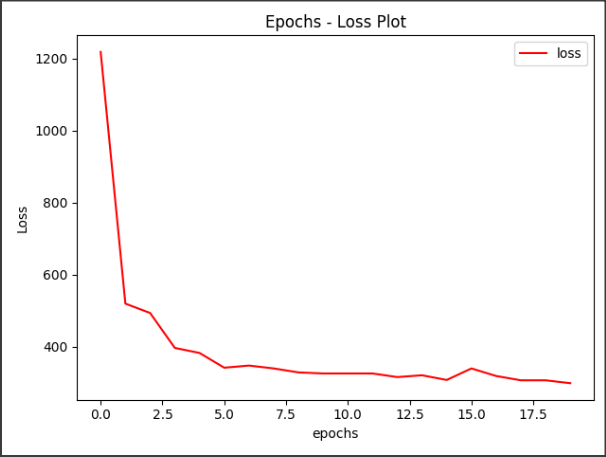
In the training model stage, the stochastic gradient descent (SGD) (9) of Keras in the TensorFlow framework was used to build the U-Net network model. And the loss function is mean squared error (MSE) (10) to reflect the dispersion of the dataset.

(9)

(10)

The biggest issue in deep learning is the vanishing gradient, which is particularly serious when using saturated activation functions such as tanh and sigmod, making network converge more and more slowly. However, the ReLU (11) function is much faster by virtue of its linear, non-saturated form. And reduce the image size from 572 x 572 to 512 x 512.

(11)

1. Methods: details of the data and techniques you'll be developing and applying;

**您将开发和应用的数据和技术的细节;**

ii. Results: written descriptions of your findings, supported by figures and tables;

iii. Discussion: a more in-depth discussion of your findings, in the wider context of the existing literature.

If your project includes primary research components it is expected that you present this work in a manner appropriate to a scientific report:

1. Participant recruitment
2. Evidence that ethical procedures have been followed
3. Study design (short summary of research methods section) – including hypotheses/research question as appropriate
4. A detailed description of the procedure
5. Results of experiment
6. Analysis of results. Consider the results of your work with respect to both your own specific hypotheses/research question and wider context identified in your literature review.

**Chapter 5**

# Conclusions

The results from this project indicate that ...

**Chapter 6**

# Reflective Analysis

The project went well ...

# References

Ascar, Davix.X., Judson, D. and Pittala, S.K. (2020) ‘SegNet Approach for Vehicle License Plate Localization.’ In: *2020 Seventh International Conference on Information Technology Trends (ITT)*, pp. 113-117, doi: 10.1109/ITT51279.2020.9320883.

Blischak, J.D., Davenport, E.R. and Wilson, G. (2016) A Quick Introduction to Version Control with Git and GitHub F. Ouellette (ed.). *PLOS Computational Biology*, 12(1) e1004668.

Chen, P., Bai, X. and Liu, W. (2014) Vehicle Color Recognition on Urban Road by Feature Context. *IEEE Transactions on Intelligent Transportation Systems*, 15(5) 2340–2346.

Chowdhury, P.N., Shivakumara, P., Raghavendra, R., Pal, U., Lu, T. and Blumenstein, M. (2020) A New U-Net Based License Plate Enhancement Model in Night and Day Images. *Lecture Notes in Computer Science*, 749–763.

Glasmachers, T. (2017) Limits of End-to-End Learning. *arXiv:1704.08305 [cs, stat]*, Available from: <https://arxiv.org/abs/1704.08305> [Accessed 15 April 2021].

detectRecog (2020) *CCPD (Chinese City Parking Dataset, ECCV)* Available from: <https://github.com/detectRecog/CCPD> [Accessed 8 February 2021].

Goodfellow, I.J., Bulatov, Y., Ibarz, J., Arnoud, S. and Shet, V. (2014) Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. *arXiv:1312.6082 [cs].*

Maglad, K. (2012) A Vehicle License Plate Detection and Recognition System. *Journal of Computer Science*, 8(3) 310–315.

Mithe, R., Indalkar, S. and Divekar, N. (2013) Optical character recognition. *Journal of Information & Communication Technology (IJRTE), 2(1), 72-75*

Islam, N., Islam, Z. and Noor, N. (2016) A Survey on Optical Character Recognition System. *ArXiv*, 10(2). Available from: <https://www.semanticscholar.org/paper/A-Survey-on-Optical-Character-Recognition-System-Islam-Islam/6bdb84c02fd56c209c893efda7df083c9ae5c4df> [Accessed 10 February 2021].

Ronneberger, O., Fischer, P. and Brox, T. (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lecture Notes in Computer Science*, 234–241.

Xu, Z., Yang, W., Meng, A., Lu, N., Huang, H., Ying, C. and Huang, L. (2018) ‘Towards End-to-End License Plate Detection and Recognition: A Large Dataset and Baseline’.In: *Proceedings of the European conference on computer vision* (ECCV) pp. 255-271.

Qin, S. and Liu, S. (2020) Towards End-to-end Car License Plate Location and Recognition in Unconstrained Scenarios. *arXiv:2008.10916 [cs]*, Available from https://arxiv.org/abs/2008.10916v1 [accessed 16 April 2021].

Siqi, S., Nanting, L., Yanjun, M. and Liping, Z. (2020) Robust Recognition of Truck License Plate in Mine Environment. *IEEE Xplore*, 36–42. Available from: <https://ieeexplore.ieee.org/abstract/document/9361802> [Accessed 19 March 2021].

Zhong, Y., Liu, Y., Luo, F. and Zhang, H. (2020) A Novel Integrated Neural Network for License Plate Detection And Recognition. *2020 Chinese Automation Congress (CAC)*, Available from: <https://ieeexplore.ieee.org/abstract/document/9326612> [Accessed 12 March 2021].

Jun-Wei Hsieh, Shih-Hao Yu, & Yung-Sheng Chen. (2002). Morphology-based license plate detection from complex scenes. *Object Recognition Supported by User Interaction for Service Robots*. doi:10.1109/icpr.2002.1047823

Koval, V., Turchenko, V., Kochan, V., Sachenko, A. and Markowsky, G. (2003) Smart license plate recognition system based on image processing using neural network. *Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2003. Proceedings*. doi:10.1109/idaacs.20031249531

Ghasrodashti, E.K. and Yazdi, M. (2010) Recognizing Persian license plates in digital zoom condition. *2010 2nd International Conference on Education Technology and Computer*, pp. V3-208-V3-212, doi: 10.1109/ICETC.2010.5529561.

Park, H., Lee, D., Lim, M., Kang, Y., Oh, J. and Kim, J.-H. (2018) A Fast-Converged Acoustic Modeling for Korean Speech Recognition: A Preliminary Study on Time Delay Neural Network. *arXiv:1807.05855 [cs, eess]*, Available from: <https://arxiv.org/abs/1807.05855> [Accessed 15 April 2021].

Rachmadi, R.F. and Purnama, I.K.E. (2018) Vehicle Color Recognition using Convolutional Neural Network. *arXiv:1510.07391 [cs]*, Available from: <https://arxiv.org/abs/1510.07391> [Accessed 9 March 2021].

Sarkar, D., Bali, R. and Sharma, T. (2018) *Practical Machine Learning with Python*. Berkeley, CA: Apress.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P. and Wicke, M. (2016) *TensorFlow: A System for Large-Scale Machine Learning.* USENIX.

Toljaga-Nikolic, D., Petrovic, D. and Mihic, M. (2017) How to choose the appropriate project management approach? *2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT)*, Available from: [https://sci-hub.do/https://ieeexplore.ieee.org/abstract/document/8099448/](https://sci-hub.do/https:/ieeexplore.ieee.org/abstract/document/8099448/) [accessed 22 April 2021].

Sharma, S., Sarkar, D. and Gupta, D. (2012) Agile Processes and Methodologies: A Conceptual Study. *International Journal on Computer Science and Engineering,* Available from: <https://www.yashada.org/yash/egovcii/static_pgs/TC/IJCSE12-04-05-186.pdf> [accessed 22 April 2021].

Wohlin, C., Höst, M. and Henningsson, K. (2006) Empirical Research Methods in Web and Software Engineering. *Web Engineering*, 409–430. Available from https://link.springer.com/chapter/10.1007/3-540-28218-1\_13 [accessed 30 April 2021].

DeFranco, J.F. and Laplante, P.A. (2017) A content analysis process for qualitative software engineering research. *Innovations in Systems and Software Engineering*, 13(2-3) 129–141.

Snyder, H. (2019) Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, 104(104) 333–339. Available from .

Ehmer, M. and Khan, F. (2012) A Comparative Study of White Box, Black Box and Grey Box Testing Techniques. *International Journal of Advanced Computer Science and Applications*, 3(6).

Abdussalam, A., Sun, S., Fu, M., Ullah, Y. and Ali, S. (2018) *Robust Model for Chinese License Plate Character Recognition Using Deep Learning Techniques* Available from https://www.semanticscholar.org/paper/Robust-Model-for-Chinese-License-Plate-Character-Abdussalam-Sun/8ecdf17fb1f164351950a987a92ea66fda501b19 [accessed 5 May 2021].

Long, J., Shelhamer, E. and Darrell, T. (2015b) *Fully Convolutional Networks for Semantic Segmentation* Available from <https://openaccess.thecvf.com/content_cvpr_2015/html/Long_Fully_Convolutional_Networks_2015_CVPR_paper.html> [accessed 5 May 2021].