**Real-time Automatic License Plate Recognition System**

**ABSTRACT**

We introduce a real-time Automatic License Plate Recognition system that is computationally lighter by eliminating the ROI setting step, without deteriorating recognition performance. Conventional license plate recognition systems exhibit two main problems. First, clear license plate visibility is required. Second, processing actual field data is computationally intensive and the ROI needs to be set. To overcome these problems, we performed plate localization directly on the entire image, and conducted research taking low quality license plate detection into account. We aim to recognize the license plates of cars moving at high speeds on the road as well as stationary cars using the hardware module, which is an embedded computing device.

**CHAPTER-1**

**INTRODUCTION:**

There are many license plate recognition systems these days. Most of these systems target large or clear license plates. However, license plate data collected from the actual field are not always appropriate for processing. License plate images comprising field data are often too small or unclear. In fact, field data is often blurred because vehicles are moving at high speeds or there is a lot of noise. Therefore, most of the previous systems proposed have difficulty in processing field data. Recently, studies overcoming this problem have been published, but due to the large amount of computation, they cannot be used in real-time on light, embedded devices. Our purpose is to create a system that allows realtime license plate recognition on a relatively light device with less computation. To do this, we need to develop a way to recognize relatively small license plates from high-resolution images, such as 3K or 4K. This allows you to skip the step of finding the car area, reducing the amount of computation and speeding up the system. Real-time object recognition methods known as 1-stage detectors, such as YOLO or SSD, were used in previous works for license plate detection and character recognition. Initially, we trained the model using YOLOv3 and SSD, but these were not efficient because the license plate size was too small as the input image size. However, YOLOv4 solved this problem. YOLOv4 was able to detect small license plates that were often undetectable by YOLOv3 or SSD. 2. Related work In this paper, real-time 1-stage detectors SSD and YOLO were used for license plate detection and character recognition.

Single Shot Multi Box Detector (SSD), was published by W. Liu et al. in 2016. SSD uses VGG16 as the base network due to its high-quality image classification, and then convolutional feature layers that progressively reduce in size are added to its end so that it is able to predict object at different scales by the aspect ratio. The core of SSD is predicting category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps. This model has been improved and implemented by many researchers by using ResNet or MobileNet instead of VGG16.

The first version of the YOLO algorithm to detect objects was published in 2016, named YOLOv1. Prior work on object detection repurposes classifiers to perform detection. Instead, J. Redmon et al. frame object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance. In 2017, the year after the initial YOLO release, YOLO9000 (YOLOv2) was published with various improvements, titled “YOLO9000: Better, Fast, Stronger”. In 2018, YOLOv3 was released with very slight changes. The latest version called YOLOv4 was published in 2020. YOLOv4, like YOLOv3, is a combination of up-to-date techniques to improve its performance.

**EXISTING SYSTEM:**

License plate images comprising field data are often too small or unclear. In fact, field data is often blurred because vehicles are moving at high speeds or there is a lot of noise. Therefore, most of the previous systems proposed have difficulty in processing field data. Recently, studies overcoming this problem have been published, but due to the large amount of computation, they cannot be used in real-time on light, embedded devices.

**DISADVANTAGES:**

* It doesn't generalize well when objects in the image show rare aspects of ratio.
* EfficientDet on the other hand, does detect small objects well; however it fails to do real-time detection with its two step architecture.
* The existing process of identifying and noticing the vehicle needs to be done manually. Hence, efficiency and accuracy will be low.

**PROPOSED SYSTEM:**

Our purpose is to create a system that allows realtime license plate recognition on a relatively light device with less computation. To do this, we need to develop a way to recognize relatively small license plates from high-resolution images, such as 3K or 4K. This allows you to skip the step of finding the car area, reducing the amount of computation and speeding up the system. Real-time object recognition methods known as 1-stage detectors, such as YOLO or SSD, were used in previous works for license plate detection and character recognition. Initially, we trained the model using YOLOv3 and SSD, but these were not efficient because the license plate size was too small as the input image size. However, YOLOv4 solved this problem. YOLOv4 was able to detect small license plates that were often undetectable by YOLOv3 or SSD.

**ADVANTAGES:**

* YOLOv4 is twice as fast as EfficientDet (competitive recognition model) with comparable performance.
* Fast recognition of a vehicle number plate is the basis for fast and seamless vehicle identification. The identification can be used to grant vehicles access or find and track specific vehicles.
* The automated recognition of number plates allows automated alerts and controls for facilities. Hence, ANPR is a key technology for smart cities.
* The precise and fast number plate recognition doesn’t rely on human input. Hence it drives cost-efficient governance and reduces waiting times.

**SYSTEM SPECIFICATION**

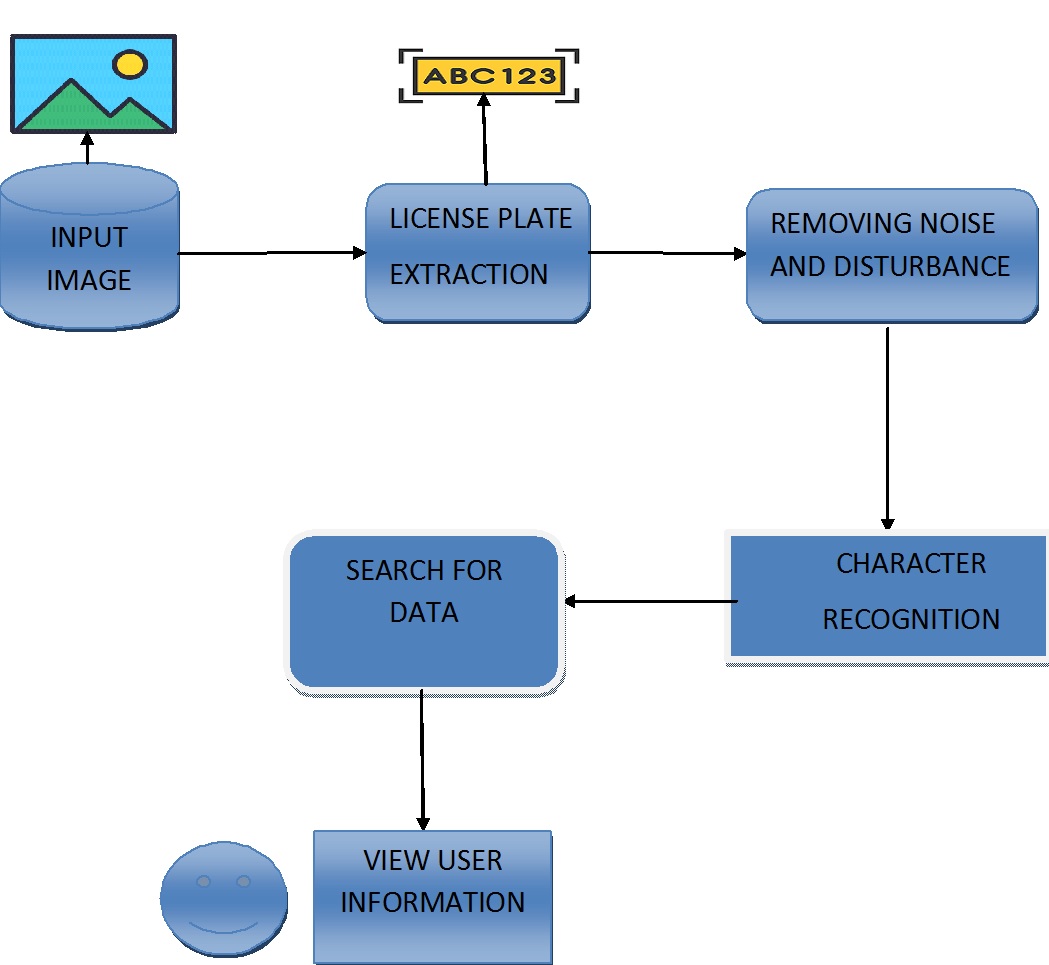
**HARDWARE CONFIGURATION:**

* Processor - I5
* Speed - 2.5 GHz
* RAM - 4 GB(min)
* Hard Disk - 500 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - LCD

**SOFTWARE CONFIGURATION**

* Operating System - Linux, Windows/7/10
* Server - Anaconda, Jupyter,pycharm
* Front End - tkinter |GUI toolkit
* Server side Script - Python , AIML

**SYSTEM ARCHITECTURE:**

****

**DATA FLOW DIAGRAM:**

VIEW DATA

SEARCH FOR DATA

INPUT IMAGE

IF VALID

LOGIN

REMOVING NOISE

REGISTER

NO

YES

CHARACTER RECOGNITION

LICENSE PLATE EXTRACTION

**USECASE DIAGRAM:**

OFFICER

SYSTEM

**ACTIVITY DIAGRAM:**

LOGIN

REGISTER

INPUT IMAGE

CHARACTER RECOGNITION

REMOVE NOISE

VIEW USER DETAILS

LICENSE PLATE EXTRACTION

**CLASS DIAGRAM:**

SYSTEM

LICENSE PLATE EXTRACT

REMOVING NOISE

CHARACTER RECOGNITION

OFFICER

INPUT IMAGE

VIEW EXTRACTED TEXT

VIEW DETAILS

**CHAPTER-2**

**LITERATURE SURVEY:**

# TITLE: Automatic License Plate Recognition System for Bangla License Plates using Convolutional Neural Network.

**YEAR OF PUBLISHING**: 2019.

**AUTHOR NAME:** [Nazmus Saif](https://ieeexplore.ieee.org/author/37087121757), [Nazir Ahmmed](https://ieeexplore.ieee.org/author/37087123551)

# ABSTRACT:

# We present an automatic license plate recognition system that can detect and recognize Bangla license plates from an image of vehicles. It is mandatory for vehicles in Bangladesh to have Bangladesh Road Transport Authority (BRTA) standard license plates attached in front and back of the vehicle. We build a database containing vehicles of this type. From this dataset, detecting the license plates in an input image is the first step of our system. We use a convolutional neural network (CNN) model for this. From that, we take the detected license plate image as a new input for the second and similar CNN model to segment and recognize license plate numbers. We tested our model on 200 images and achieved an accuracy of 99.5%. For speed, we tested our model on Google Collaborator’s Nvidia Tesla K80 GPU and attained a speed of 9 frames per second while detecting and recognizing the license plate numbers in a video.

# TITLE: An Analytical Approach for Enhancing the Automatic Detection and Recognition of Skewed Bangla License Plates.

**YEAR OF PUBLISHING**: 2020

**AUTHOR NAME:** [Koushik Roy](https://ieeexplore.ieee.org/author/37086839037),[Abu Mohammad Shabbir Khan](https://ieeexplore.ieee.org/author/37088393804)

# ABSTRACT:

# Although there has been a huge body of work on Bangla license plate detection and recognition, the successes of these works have largely been limited to correct detection and recognition of undistorted license plates whose images are taken chiefly from the front or the back of vehicles with slight angular variations. As a result, most Bangla automatic license plate recognition (ALPR) systems in practice struggle when the license plates are skewed on the viewing or the image planes of the license plates. In this paper, we address this issue by proposing an analytical approach that can enhance the ALPR of both normal and skewed license plates and can be incorporated into existing Bangla ALPR systems without modifying their internal structures. Specifically, we demonstrate how existing ALPR systems can be treated as black boxes and analyzed to understand what sort of license plate images they work best on and introduce a novel pipeline that combines deep learning and an algorithmic procedure for transforming images of both normal and skewed license plates into formats that are best suited for the ALPR systems. We note that our proposed method can be easily generalized and applied to non-Bangla license plates as well.

# TITLE: Real-Time Bangla License Plate Recognition System using Faster R-CNN and SSD: A Deep Learning Application

**YEAR OF PUBLISHING**: 2020.

**AUTHOR NAME:** [Tariqul Islam](https://ieeexplore.ieee.org/author/37089128062),[Risul Islam Rasel](https://ieeexplore.ieee.org/author/37072497600)

**ABSTRACT:**

Traffic control and vehicle owner identification become major problems in Bangladesh. Most of the time it is difficult to identify the driver or the owner of the vehicles who violate the traffic rules or do any accidental work on the road. Moreover, it is very time-consuming for a traffic police officer to physically check the license plate of every vehicle. So, an automatic license plate recognition system is a much-needed solution to solve these problems. The existing Bangla license plate recognition systems are mostly based on character segmentation and these methods are not implemented in real-time. In this study, two separate Deep Convolutional Neural Network (DCNN) models are used to identify the license plate and the characters on the license plate from the real-time video streaming. The first CNN model detects the license plate from the live video of a vehicle on the road. Than it crop the license plate area from the video frames. The cropped frame is then fed into the second CNN to detect the characters on that license plate. The characters are detected as individual objects. After detecting all the characters and numbers on the license plate, they are rearranged according to their position on the plate. To train the proposed model total of 292 images are collected used. Moreover, an open-sourced Bangla handwritten character dataset named BanglaLekha-Isolated is also used to train the model with synthetic character data. The trained model is tested using 18 live videos and 6 still image data. Finally, the proposed methodology gains a 100% precision on detecting the license plate, and 91.67% precision for detecting the characters on the license plate for the given test dataset.

# TITLE: Automatic License Plate Detection System for Myanmar Vehicle License Plates.

**YEAR OF PUBLISHING**: 2019.

**AUTHOR NAME:** [Khin Pa Pa Aung](https://ieeexplore.ieee.org/author/37087114694),[Khin Htar Nwe](https://ieeexplore.ieee.org/author/37087115419)

# ABSTRACT:

License Plate Detection is an intelligent system to find the exact license plate by analyzing image/video data for automatic license plate recognition (ALPR) system. The proposed system used image data. There is much early research has been done for ALPR purposes, however, it was still challenging tasks for accurately detect license plates in the open environment. The main difficulties lie in the diversity of plates such as language, font, color, and type of the number plates that differ across nations and conditional variations such as various background scenes and illumination when captured. The fundamental ALPR system consists of three processes: license plate detection, character segmentation, and character recognition. Proposed system focused on license plate detection based on image processing technology that is a crucial step for the whole ALPR system. A Myanmar License plate has a white boundary for every different color of plates, thus the proposed system applied an edge-based approach for any color of plates, and the plate region will remain because of its white boundary. After edge detection, morphological operation has been applied as it can add or remove pixels from/to the objects in an image. Thus the license plate can be extracted accurately. Finally, the bounding box technology was applied to extract only the number plate region properly.

# TITLE: Robust Real time Lightweight Automatic License plate Recognition System for Iranian License Plates.

**YEAR OF PUBLISHING**: 2020

**AUTHOR NAME:** [Yusef Alborzi](https://ieeexplore.ieee.org/author/37088376407), Talayeh Sarraf Mehraban

# ABSTRACT:

In this paper we propose an Automatic License Plate Recognition (ALPR) system for unsupervised parking lot applications. The main objective is to develop a system which is implementable on embedded devices, specifically a Raspberry-PI3. ALPR consists of two main stages: (I) Locating the plate and (II) Optical Character Recognition (OCR). Considering the recent growth and success of deep learning methods, especially convolutional neural networks (CNN), in our system, we used the Single Shot Detection (SSD) architecture along with the MobileNet feature extractor to detect the plate in the image captured by the camera and the LPRNet network for OCR. The proposed method is robust, accurate, computationally inexpensive and able to perform in Real-time. The system achieves 79.86% end-to-end accuracy on our dataset and successfully performs in real-time on a Raspberry-PI3. For training the OCR network, we generated and used 130k synthetic license plate images. We also introduce a dataset containing 1500 images with various camera zoom, lighting and viewpoint conditions.

**CHAPTER 3 :MODULE DESCRIPTION**

**MODULES**

1. Input Images
2. License Plate Extraction
3. Removing Noise
4. Character Recognition

**MODULE DESCRIPTION**

**1. Input Images:**

In this module input the specified image from the control using the Image Location property and is the action of retrieving an image from a source.

**2. License Plate Extraction:**

Licenseplate extraction is real life research area in the field of image processing for automatic vehicle identification. In this module the license plate extract from the image we load previously using the yolov4.

**3. Removing Noise:**

With the basis of color combination and text based technique the noise and unwanted characters were removed to attain accuracy in text detection..

**4. Character Recognition:**

After removal of noise and unwanted content it makes easier to identifies the text exactly and then subjecting into analyzing of reading text, with the used of curve based techniques the selected content were compared and text were identified as readable and editable.

**CHAPTER-4**

**DOMAIN OF THE PROJECT**

**PYTHON:**

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding; make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.It ranges from simple automation tasks to gaming, web development, and even complex enterprise systems. These are the areas where this technology is still the king with no or little competence: Machine learning as it has a plethora of libraries implementing machine learning algorithms.Python is a one-stop shop and relatively easy to learn, thus quite popular now. What other reasons exist for such universal popularity of this programming language and what companies have leveraged its opportunities to the max? Let’s talk about that. Python technology is quite popular among programmers, but the practice shows that business owners are also Python development believers and for good reason. Software developers love it for its straightforward syntax and reputation as one of the easiest programming languages to learn. Business owners or CTOs appreciate the fact that there’s a framework for pretty much anything – from web apps to machine learning. Moreover, it is not just a language but more a technology platform that has come together through a gigantic collaboration from thousands of individual professional developers forming a huge and peculiar community of aficionados. So what is python used for and what are the tangible benefits the language brings to those who decided to use it? Below we’re going to discover that. Productivity and Speed It is a widespread theory within development circles that developing Python applications is approximately up to 10 times faster than developing the same application in Java or C/C++. The impressive benefit in terms of time saving can be explained by the clean object-oriented design, enhanced process control capabilities, and strong integration and text processing capacities. Moreover, its own unit testing framework contributes substantially to its speed and productivity.

**PYCHARM**

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

Choose the best PyCharm for you﻿

**PyCharm is available in three editions:**

* Community (free and open-sourced): for smart and intelligent Python development, including code assistance, refactorings, visual debugging, and version control integration.
* Professional (paid) : for professional Python, web, and data science development, including code assistance, refactorings, visual debugging, version control integration, remote configurations, deployment, support for popular web frameworks, such as Django and Flask, database support, scientific tools (including Jupyter notebook support), big data tools.
* Edu (free and open-sourced): for learning programming languages and related technologies with integrated educational tools.

For details, see the editions comparison matrix.

**Supported languages﻿**

To start developing in Python with PyCharm you need to download and install Python from python.org depending on your platform.

PyCharm supports the following versions of Python:

Python 2: version 2.7

Python 3: from the version 3.6 up to the version 3.10

Besides, in the Professional edition, one can develop Django, Flask, and Pyramid applications. Also, it fully supports HTML (including HTML5), CSS, JavaScript, and XML: these languages are bundled in the IDE via plugins and are switched on for you by default. Support for the other languages and frameworks can also be added via plugins (go to Settings | Plugins or PyCharm | Preferences | Plugins for macOS users, to find out more or set them up during the first IDE launch).

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Minimum** | **Recommended** |
| RAM | 4 GB of free RAM | 8 GB of total system RAM |
| CPU | Any modern CPU | Multi-core CPU. PyCharm supports multithreading for different operations and processes making it faster the more CPU cores it can use. |
| Disk space | 2.5 GB and another 1 GB for caches | SSD drive with at least 5 GB of free space |
| Monitor resolution | 1024x768 | 1920×1080 |
| Operating system | Officially released 64-bit versions of the following:   * Microsoft Windows 8 or later * macOS 10.13 or later * Any Linux distribution that supports Gnome, KDE, or Unity DE. PyCharm is not available for some Linux distributions, such as RHEL6 or CentOS6, that do not include [GLIBC](https://ftp.gnu.org/gnu/libc/) 2.14 or later.   Pre-release versions are not supported. | Latest 64-bit version of Windows, macOS, or Linux (for example, Debian, Ubuntu, or RHEL) |

**SPYDER**

**Spyder** is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open-source software. It is released under the MIT license.

Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community.

Spyder is extensible with first-party and third-party plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, [Pylint](https://en.wikipedia.org/wiki/Pylint) and Rope. It is available cross-platform through [Anaconda](https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)), on Windows, on macOS through [MacPorts](https://en.wikipedia.org/wiki/MacPorts), and on major Linux distributions such as [Arch Linux](https://en.wikipedia.org/wiki/Arch_Linux), [Debian](https://en.wikipedia.org/wiki/Debian), [Fedora](https://en.wikipedia.org/wiki/Fedora_(operating_system)), [Gentoo Linux](https://en.wikipedia.org/wiki/Gentoo_Linux), [openSUSE](https://en.wikipedia.org/wiki/OpenSUSE) and [Ubuntu](https://en.wikipedia.org/wiki/Ubuntu_(operating_system)).

Spyder uses [Qt](https://en.wikipedia.org/wiki/Qt_(software)) for its GUI and is designed to use either of the [PyQt](https://en.wikipedia.org/wiki/PyQt) or [PySide](https://en.wikipedia.org/wiki/PySide) Python bindings. QtPy, a thin abstraction layer developed by the Spyder project and later adopted by multiple other packages, provides the flexibility to use either backend.

**FEATURES**

Features include:

* An editor with [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), [introspection](https://en.wikipedia.org/wiki/Introspection_(computer_science)), [code completion](https://en.wikipedia.org/wiki/Code_completion)
* Support for multiple [IPython](https://en.wikipedia.org/wiki/IPython) [consoles](https://en.wikipedia.org/wiki/Command-line_interface)
* The ability to explore and edit [variables](https://en.wikipedia.org/wiki/Variable_(computer_science)) from a [GUI](https://en.wikipedia.org/wiki/Graphical_user_interface)
* A Help pane able to retrieve and render rich text [documentation](https://en.wikipedia.org/wiki/Application_programming_interface#Documentation) on functions, classes and methods automatically or on-demand
* A [debugger](https://en.wikipedia.org/wiki/Debugger) linked to IPdb, for step-by-step execution
* [Static code analysis](https://en.wikipedia.org/wiki/Static_program_analysis), powered by [Pylint](https://en.wikipedia.org/wiki/Pylint)
* A run-time [Profiler](https://en.wikipedia.org/wiki/Profiling_(computer_programming)), to benchmark code
* Project support, allowing work on multiple development efforts simultaneously
* A built-in [file explorer](https://en.wikipedia.org/wiki/File_manager), for interacting with the file system and managing projects
* A "Find in Files" feature, allowing full [regular expression](https://en.wikipedia.org/wiki/Regular_expression) search over a specified scope
* An online help browser, allowing users to search and view Python and package documentation inside the IDE
* A [history log](https://en.wikipedia.org/wiki/Command_history), recording every user command entered in each console
* An internal console, allowing for introspection and control over Spyder's own operation

**PLUGINS**

Available plugins include:

* Spyder-Unittest, which integrates the popular [unit testing](https://en.wikipedia.org/wiki/Unit_testing) frameworks Pytest, Unittest and Nose with Spyder
* Spyder-Notebook, allowing the viewing and editing of [Jupyter Notebooks](https://en.wikipedia.org/wiki/IPython#Project_Jupyter) within the IDE
* Download Spyder Notebook
* Using conda: *conda install spyder-notebook -c spyder-ide*
* Using pip: *pip install spyder-notebook*
* Spyder-Reports, enabling use of [literate programming](https://en.wikipedia.org/wiki/Literate_programming) techniques in Python
* Spyder-Terminal, adding the ability to open, control and manage cross-platform [system shells](https://en.wikipedia.org/wiki/Shell_(computing)) within Spyder
  + Download Spyder Terminal
  + Using conda: *conda install spyder-terminal -c spyder-ide*
  + Using pip: *pip install spyder-terminal*
* Spyder-Vim, containing commands and shortcuts emulating the [Vim text editor](https://en.wikipedia.org/wiki/Vim_(text_editor))
* Spyder-AutoPEP8, which can automatically conform code to the standard PEP 8 [code style](https://en.wikipedia.org/wiki/Programming_style)
* Spyder-Line-Profiler and Spyder-Memory-Profiler, extending the built-in profiling functionality to include testing an individual line, and measuring [memory](https://en.wikipedia.org/wiki/Random_access_memory) usage

**ANACONDA PYTHON**

Anaconda® is a package manager, an environment manager, a Python/R data science distribution, and a collection of [over 7,500+ open-source packages](https://docs.anaconda.com/anaconda/packages/pkg-docs/). Anaconda is free and easy to install, and it offers [free community support](https://groups.google.com/a/anaconda.com/forum/?fromgroups#!forum/anaconda).

Get the Anaconda Cheat Sheet and then [download Anaconda](https://www.anaconda.com/downloads).

Want to install conda and use conda to install just the packages you need? Get [Miniconda](http://conda.pydata.org/miniconda.html).

**Anaconda Navigator or conda?**

After you install Anaconda or Miniconda, if you prefer a desktop graphical user interface (GUI) then use [Navigator](https://docs.anaconda.com/anaconda/navigator/). If you prefer to use Anaconda prompt (or terminal on Linux or macOS), then use that and conda. You can also switch between them.

You can install, remove, or update any Anaconda package with a few clicks in Navigator, or with a single conda command in Anaconda Prompt (terminal on Linux or macOS).

* **To try Navigator**, after installing Anaconda, click the Navigator icon on your operating system’s program menu, or in Anaconda prompt (or terminal on Linux or macOS), run the command anaconda-navigator.
* **To try conda**, after installing Anaconda or Miniconda, take the [20-minute conda test drive](https://conda.io/projects/conda/en/latest/user-guide/getting-started.html) and download a [conda cheat sheet](https://docs.conda.io/projects/conda/en/latest/user-guide/cheatsheet.html).

**Packages available in Anaconda**

* Over [250 packages](https://docs.anaconda.com/anaconda/packages/pkg-docs/) are automatically installed with Anaconda.
* Over 7,500 additional open-source packages (including R) can be individually installed from the Anaconda repository with the conda install command.
* Thousands of other packages are available from [Anaconda.org](https://anaconda.org/).
* You can download other packages using the pip install command that is installed with Anaconda. [Pip packages](https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-pkgs.html#installing-non-conda-packages) provide many of the features of conda packages and in some cases they can work together. However, the preference should be to install the conda package if it is available.
* You can also make your own [custom packages](https://conda.io/projects/conda-build/en/latest/) using the conda build command, and you can share them with others by uploading them to [Anaconda.org](http://anaconda.org/), PyPI, or other repositories.

**Previous versions**

Previous versions of Anaconda are available in the [archive](https://repo.anaconda.com/archive/). For a list of packages included in each previous version, see [Old package lists](https://docs.anaconda.com/anaconda/packages/oldpkglists/).

Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it does not matter which one you download, because you can create new environments that include any version of Python packaged with conda. See [Managing Python with conda](https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-python.html).

**tkinter – Python**

Tk/Tcl has long been an integral part of Python. It provides a robust and platform independent windowing toolkit, that is available to Python programmers using the [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) package, and its extension, the [tkinter.tix](https://docs.python.org/3/library/tkinter.tix.html#module-tkinter.tix) and the [tkinter.ttk](https://docs.python.org/3/library/tkinter.ttk.html#module-tkinter.ttk) modules.

The [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) package is a thin object-oriented layer on top of Tcl/Tk. To use [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter), you don’t need to write Tcl code, but you will need to consult the Tk documentation, and occasionally the Tcl documentation. [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) is a set of wrappers that implement the Tk widgets as Python classes.

[tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter)’s chief virtues are that it is fast, and that it usually comes bundled with Python. Although its standard documentation is weak, good material is available, which includes: references, tutorials, a book and others. [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) is also famous for having an outdated look and feel, which has been vastly improved in Tk 8.5. Nevertheless, there are many other GUI libraries that you could be interested in. The Python wiki lists several alternative [GUI frameworks and tools](https://wiki.python.org/moin/GuiProgramming).

**Main tkinter module.**

[tkinter.colorchooser](https://docs.python.org/3/library/tkinter.colorchooser.html#module-tkinter.colorchooser)

**Dialog to let the user choose a color.**

[tkinter.commondialog](https://docs.python.org/3/library/dialog.html#module-tkinter.commondialog)

**Base class for the dialogs defined in the other modules listed here.**

[tkinter.filedialog](https://docs.python.org/3/library/dialog.html#module-tkinter.filedialog)

**Common dialogs to allow the user to specify a file to open or save.**

[tkinter.font](https://docs.python.org/3/library/tkinter.font.html#module-tkinter.font)

**Utilities to help work with fonts.**

[tkinter.messagebox](https://docs.python.org/3/library/tkinter.messagebox.html#module-tkinter.messagebox)

**Access to standard tk dialog boxes.**

[tkinter.scrolledtext](https://docs.python.org/3/library/tkinter.scrolledtext.html#module-tkinter.scrolledtext)

**Text widget with a vertical scroll bar built in.**

[tkinter.simpledialog](https://docs.python.org/3/library/dialog.html#module-tkinter.simpledialog)

**Basic dialogs and convenience functions.**

[tkinter.ttk](https://docs.python.org/3/library/tkinter.ttk.html#module-tkinter.ttk)

Themed widget set introduced in Tk 8.5, providing modern alternatives for many of the classic widgets in the main [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) module.

**Additional modules:**

\_tkinter

A binary module that contains the low-level interface to Tcl/Tk. It is automatically imported by the main [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) module, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

idlelib

Python’s Integrated Development and Learning Environment (IDLE).Based on [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter).

tkinter.constants

Symbolic constants that can be used in place of strings when passing various parameters to Tkinter calls.Automatically imported by the main [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) module.

[tkinter.dnd](https://docs.python.org/3/library/tkinter.dnd.html#module-tkinter.dnd)

(experimental) Drag-and-drop support for [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter). This will become deprecated when it is replaced with the Tk DND.

[tkinter.tix](https://docs.python.org/3/library/tkinter.tix.html#module-tkinter.tix)

(deprecated) An older third-party Tcl/Tk package that adds several new widgets. Better alternatives for most can be found in [tkinter.ttk](https://docs.python.org/3/library/tkinter.ttk.html#module-tkinter.ttk).

[turtle](https://docs.python.org/3/library/turtle.html#module-turtle)

Turtle graphics in a Tk window.

**CHAPTER-5**

**SYSTEM TESTING AND MAINTENANCE:**

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. In the testing process we test the actual system in an organization and gather errors from the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed to ensuring that the system works accurately and efficiently.

In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently.

The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So we should have more conscious to give input data. It is important to give correct inputs to efficient testing.

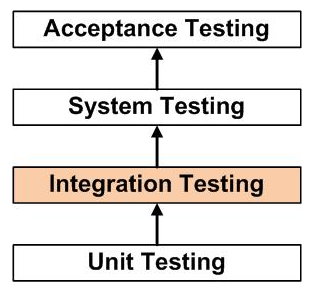
Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works. Inadequate testing or non-testing leads to errors that may appear few months later.

This will create two problems, Time delay between the cause and appearance of the problem. The effect of the system errors on files and records within the system. The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the system to its limits.

The testing process focuses on logical intervals of the software ensuring that all the statements have been tested and on the function intervals (i.e.,) conducting tests to uncover errors and ensure that defined inputs will produce actual results that agree with the required results. Testing has to be done using the two common steps Unit testing and Integration testing. In the project system testing is made as follows:

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every months or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and reports.

Integration Testing is a level of software testing where individual units are combined and tested as a group.



The purpose of this level is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration testing.

**METHOD**

Any of [Black Box Testing](http://softwaretestingfundamentals.com/black-box-testing/), [White Box Testing](http://softwaretestingfundamentals.com/white-box-testing/), and [Gray Box Testing](http://softwaretestingfundamentals.com/gray-box-testing/) methods can be used. Normally, the method depends on your definition of ‘unit’.

**TASKS**

* Integration Test Plan
  + Prepare
  + Review
  + Rework
  + Baseline
* Integration Test Cases/Scripts
  + Prepare
  + Review
  + Rework
  + Baseline
* Integration Test
  + Perform

**UNIT TESTING:**

Unit testing verification efforts on the smallest unit of software design, module. This is known as “Module Testing”. The modules are tested separately. This testing is carried out during programming stage itself. In these testing steps, each module is found to be working satisfactorily as regard to the expected output from the module.

**BLACK BOX TESTING**

**Black box testing,** also known as Behavioral Testing, is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

**WHITE-BOX TESTING**

**White-box testing** (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software thattests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing).

**GREY BOX TESTING**

**Grey box testing** is a technique to test the application with having a limited knowledge of the internal workings of an application. To test the Web Services application usually the Grey box testing is used. Grey box testing is performed by end-users and also by testers and developers.

**INTEGRATION TESTING:**

Integration testing is a systematic technique for constructing tests to uncover error associated within the interface. In the project, all the modules are combined and then the entire programmer is tested as a whole. In the integration-testing step, all the error uncovered is corrected for the next testing steps.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**ACCEPTANCE TESTING**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Acceptance testing for Data Synchronization:

* The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
* The Route add operation is done only when there is a Route request in need
* The Status of Nodes information is done automatically in the Cache Updating process

**BUILD THE TEST PLAN**

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

**CHAPTER-6**

**CONCLUSION**

We propose a fast and accurate automatic license plate recognition system that is suitable for processing field data. By enabling small detection in the image, the burden of setting the ROI area was relieved, reducing the amount of computation and increasing the speed. In addition, the performance is improved by using YOLOv4 detector that can recognize some contorted characters. As future work, we propose to increase the character recognition dataset. Since most of the datasets were recently photographed in Seoul, the number of old license plates, regional license plates, and unusual character datasets is less than that of general numeric datasets. Addition of rare license plates to the dataset can improve the performance of the training model.

**CHAPTER-7**

**REFERENCES:**

[1] R. Laroca, E. Severo, L. A. Zanlorensi, L. S. Oliveira, G. R. Gonçalves, W. R. Schwartz, and D. Menotti, “A robust real-time automatic license plate recognition based on the YOLO detector,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), 2018, pp. 1–10.

[2] O. Bulan, V. Kozitsky, P. Ramesh, and M. Shreve, “Segmentation- and annotation-free license plate recognition with deep localization and failure identification,” IEEE Trans. Intell. Transp. Syst., vol. 18, no. 9, pp. 2351–2363, Sep. 2017.

[3] R.-C. Chen, et al., Automatic license plate recognition via sliding-window darknet-YOLO deep learning, Image Vis. Comput. 87 (2019) 47–56.

[4] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2016, pp. 779–788.