VCS – Vehicle Counting System

PROJECT REPORT

Submitted by

Saviyo Thomas

Reg.No: 182113328

In partial fulfilment for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

MAHATMA GANDHI UNIVERSITY

CMS COLLEGE, KOTTAYAM

(AUTONOMOUS)

CMS COLLEGE, KOTTAYAM

(AUTONOMOUS)

BONAFIDE CERTIFICATE

Certified that project work entitled “ VCS – Vehicle Counting System“ is a Bonafede work carried out in the sixth semester by **Saviyo Thomas** in partial fulfilment for the award of Bachelor of Computer application from CMS College, Kottayam during the academic year 2020-2021.

SIGNATURE OF GUIDE	**SIGNATURE OF SEMINAR CO-ORDINATOR**
Ms. Naveena Tresa Joseph	**Ms. Athiramol S**
Assistant Professor	**Assistant Professor**

SIGNATURE OF HEAD OF THE DEPARTMENT in charge

Ms. Delsey MJ

Assistant Professor

CMS COLLEGE, KOTTAYAM (AUTONOMOUS)

CERTIFICATE

Certificate that the candidate **Saviyo Thomas** was examined for the seminar work entitled

” VCS – Vehicle Counting System” and the seminar was conducted on

Internal Examiner: **External Examiner:**

Date: **Date:**

DECLARATION

I **Saviyo Thomas**, do hereby declare that this project work entitled ”VCS – Vehicle Counting System” submitted to CMS COLLEGE(AUTONOMOUS), Kottayam in partial fulfilment of the requirements for the award of Bachelor of Computer Application, is a record of original work done by me. I also promise that this work has not been submitted to any other university or institution for the award of any Degree.

Name of the candidate	**: Saviyo Thomas**
Registered Number	**: 182113328**
Signature	**:**
Place	**: Kottayam**
Date	**:**

Acknowledgement

Dedicating this project to the Almighty God whose abundant grace and mercies enabled its successful completion, we would like to express our profound gratitude to all the people who had inspired and motivated us to make this project a success.

We would express our gratitude to the management and the computer department of CMS College KOTTAYAM for advanced studies for providing us with all the required facilities without which the project would not have be possible. We express our heart full gratitude to our Principal, **Dr Varghese C Joshua** for his warm support with regard to our work.

We would express our deep sense of gratitude to **Assistant Prof. Delsey M J** (Head of the Department) for her valuable help and guidance during the course of the project.

We would express our deep sense of gratitude to **Assistant Prof. Ms. Naveena Tresa Joseph** for her valuable help and guidance during the course of the project.

We are deeply indebted to our project coordinator **Mrs.ATHIRAMOL S** for providing us with valuable advice and guidance during the course of the project.

We also extend our sincere thanks to all other members for the facility of the department of Computer Application for their assistance and encouragement throughout the project. Last, but not least I would like to thank my friends for their co-operations and encouragement.

Place: **Saviyo Thomas**

Date:

Contents

Abstract

The project titled “VCS – Vehicle Counting System” is a python script. The main objective of developing this project is to help users to better analyse traffic cam footage and derive data from it. User just needs to provide a video file directory to the system and it displays the output video with real-time counting of vehicles passing through. And a text file is generated for later references. This script auto mates the task and is very reliable.

List of Figures

List of Tables

Chapter 1

Introduction

The project VCS is developed with the intent to help users access analyzed data from traffic cam footage. This script will greatly mitigate the manual work of the user. It handles the objects moving in the subject video and records it if it crosses a line drawn in the frame across the road. The user inputs the video file and the time and date information of when the video is from, then the script analyses the video and creates a foreground mask using absolute difference of the frames and then uses centroid tracking algorithm to track the objects to the line above mentioned then the objects are recorded into the list if the centroid crosses the line, then as the script is closing the plotter module enters the vehicle count to a text file.

1.1 Existing System

The existing systems in place makes use of xml files or just manual counting. These techniques are not accurate enough or fast enough or easy enough.

Limitations of existing systems

- Erroneous detection
- Failure to track subject effectively
- Slow processing

1.2 Proposed System

The proposed project only requires the source of the video and the date and time when it was taken. The system is capable of generating a video output and a text file with the past records.

Advantages of proposed system

- Accurate detection
- Effective tracking
- Real time data plotting
- Faster processing

Objectives of Proposed system

- To help users access analyzed data from traffic cam footage.
- To automate the task
- To create a flexible system
- To save time and cost
- For faster and reliable results

CHAPTER 2

SYSTEM REQUIREMENT ANALYSIS

Requirements Analysis is the process of defining the expectations of the users for an application that is to be built or modified. Requirements analysis involves all the tasks that are conducted to identify the needs of different stakeholders. Therefore requirements analysis means to analyse, document, validate and manage software or system requirements. High-quality requirements are documented, actionable, measurable, testable, traceable, helps to identify business opportunities, and are defined to a facilitate system design.

2.1 Requirements analysis process

The requirements analysis process involves the following steps:

Eliciting requirements

The process of gathering requirements by communicating with the customers is known as eliciting requirements.

Analysing requirements

This step helps to determine the quality of the requirements. It involves identifying whether the requirements are unclear, incomplete, ambiguous, and contradictory. These issues resolved before moving to the next step.

Requirements modelling

In Requirements modelling, the requirements are usually documented in different formats such as use cases, user stories, natural-language documents, or process specification.

Review and retrospective

This step is conducted to reflect on the previous iterations of requirements gathering in a bid to make improvements in the process going forward.

2.2 Hardware requirements

- 3 GB RAM
- 3.00 GHZ CPU
- Windows 7 + or Linux 18.04 +

2.3 Software requirements

- Python
- Tkinter (Python Library)
- OpenCV (Python Library)

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION

Requirements analysis is a detailed study of various operations by a system and their relationship within and outside the system. One aspect is the design of boundaries if the system and whether not a candidate system should be related to other systems.

Data is collected from available files and transactions handled by the system. Common tools used in analysis are data flow diagrams, interviews and on observations.

The analysis of the user requirements takes place throughout the investigation and outline design stages of the system development but at some time it should be formally documented.

The input to the Software Requirements Specification (SERS) is inherently informal and imprecise, inconsistent and it is likely to be incomplete. Since the collected information is just imagination this phase gives shape to the problem. By correcting errors in this phase cost of the project can be reduced.

The output of the requirement specification is the Software Requirement Specification (SRS).

3.1 Definition of SRS

It is a fundamental document which

- Lists the requirements of a system
- Describes the major features of the system.
- Includes use cases and sequence diagrams to incorporate UML standards.

The recommendation would be used to provide clear visibility of the product to be developed serving as a baseline for execution of contract between client and developer.

3.2 Features of SRS

- Functionality of the software.
- External interface- How the system reacts with the users or with other systems or hardware that would invoke it.
- Performance-Speed, Availability, Response and Recovery time.
- Attributes- Portability, Correctness, Maintainability and Security considerations.
- Design Constraints- Standard to be followed, resource limits, operating environments, policies for database integrating etc.
- Non-Functional requirements- Business rules, software quality attributes, safety Requirements, security requirements and performance parameters.

Terminology used in SRS should be consistent throughout the document. Every term should represent only one meaning to avoid confusion. Language experts should be consulted to review language and grammar of the SRS.

CHAPTER 4

SYSTEM STUDY

4.1 Feasibility Study

The feasibility study is carried out to determine of which of the possible solutions, is the best that can be developed with the available resources.

There are 3 types of feasibility stud

4.1.1 Technical feasibility

The evaluation determines whether the technology required for the proposed system is available and how it can be integrated within the organization. It also considers whether the existing system can be upgraded to use the newer technology that the organization can use. It is an online system by hosting in the web.

4.1.2 Economic feasibility

The evaluation looks at the financial aspects of the project to determine whether investment needed to implement the system will be recovered. This will result in a cost-benefit analysis. If the system is developed at a reasonable cost with available hardware, software and manpower then the benefits outweigh the cost. Then it is economically feasible.

4.1.3 Operational feasibility

This covers two aspects, technical performance as in whether the system can provide the right information for the organization's personnel and whether the right information is delivered to the right person at the right time. The other aspect is whether any job reconstructing or retaining is needed to implement the system. The proposed project is beneficial because it can be turned into an information system that will meet the organization operating requirements.

4.2 About development tools

4.2.1 Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system capable of collecting reference cycles. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3. Python interpreters are available for many operating systems.

Python Features

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented programming. In Python, we don't need to declare the type of variable because it is a dynamic typed language. For example, x=10 here x can be anything such as String, int etc. There are many features in Python, some of which are discussed below –

1. Easy to code: Python is high level programming language. Python is very easy to learn language as compared to other language like c, JavaScript, java etc. It is very easy to code in python 10 language and anybody can learn python basic in few hours or days .It is also developer friendly language.

2. Free and Open Source: Python language is freely available .Since, it is open-source, and this means that source code is also available to the public. So you can download it as, use it as well as share it.

3. Object-Oriented Language: One of the key features of python is Object-Oriented programming. Python supports object oriented language and concepts of classes, objects encapsulation etc.

4. GUI Programming Support: Graphical Users interfaces can be made using a module such as PyQt5, PyQt4, wx Python or Tk in python. PyQt5 is the most popular option for creating graphical apps with Python.

5. High-Level Language: Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

6. Extensible feature: Python is an Extensible language .we can write our some python code into c or c++ language and also we can compile that code in c/c++ language.

7. Python is Portable language: Python language is also a portable language for example, if we have python code for windows and if we want to run this code on other platform such as Linux, Unix and Mac then we do not need to change it, we can run this code on any platform.

8. Python is integrated language: Python is also an integrated language because we can easily integrated python with other language like c, c++ etc.

9. Interpreted Language: Python is an Interpreted Language, because python code is executed line by line at a time. Like other language c, c++, java etc. there is no need to compile python code this makes it easier to debug our code. The source code of python is converted into an immediate form called byte code.

10. Large Standard Library: 11 Python has a large standard library which provides rich set of module and functions so you do not have to write your own code for every single thing .There are many libraries present in python for such as regular expressions, unit-testing, web browsers etc.

11. Dynamically Typed Language: Python is dynamically-typed language. That means the type (for example- int, double, long etc.) for a variable is decided at run time not in advance. Because of this feature we don't need to specify the type of variable.

Python Syntax Basics

There are two ways in which you can write and execute a basic Python program:

1. In Interactive mode - where you write a program and execute it.

2. Using Script mode - where you execute and already saved Python program (.py file)

let's see those in action.

Example:

Writing "Hello, World!" On Interpreter To enter the interactive Python mode enter the following command on your Terminal.

```
$ Python
```

And, now you should be in the interactive mode. But, if you are using an IDE you don't need to type the command above to get into interactive programming mode. Here is Basic Python syntax for executing a Hello World Program. When you write this program and press enter, your IDE should display "Hello, Word!"

```
print (”Hello World”)
```

Anything that you write within " " of print (" ") gets printed on your IDE. In our case, the output that you should see on your screen would be Hello, World!.

Running Program in Scripting Mode: When you wrote your Hello World program earlier, let's assume that you saved it in a Python file. When you save your first program, look for a .py extension file. Assuming that you saved it as Hello_Python.py, here's how you would execute this code in a scripting mode.

First of all, the file you saved has to be made executable. We ideally do that by the command:

```
$ chmod +x test.py
```

Now that your file is executable, let's run the program in scripting mode.

```
$ python Hello_Python.py
```

Once you execute this command, you will see "Hello, World!" was printed on your Terminal. ”Hello, World” And, there you have learned your first Python Syntax. In our case, the output that you should see on your screen would be Hello, World!

4.2.2 Operating system

This project work is done in Windows 10, which is the operating system. An operating system is a set of software tools designed to make it easy for people or programmers to make optimum use of the computer. People can be separated into two groups, users and programmers. The user wants a convenient set of commands to manage files of data or programs, copy and run application packages while a programmer uses a set of tools that can be held together and debug programs. No matter where you are working, your computer will easier to use and manage, because Microsoft Windows 10 is more compatible and powerful than any workstation you have used.

The main features of Windows 10 are:

1. Easier to use

2. Easier to manage

3. More compatible

4. More powerful

1. Easier to use: With Windows 10, you can have faster access to information, and you are able to accomplish tasks more quickly and easily. Windows 10 makes it easier to:

- Work with files
- Find information.
- Personalize computing environment.
- Work remotely
- Work taking place the web

2. Easier to manage: You and your network administrators can work more efficiently now, because many of the most common management tasks are streamlined with Windows 8. With Windows 10 your workstation will be easier to:

- Setup
- Administrate
- Support

3. More compatible Windows 10 offers increased compatibility. With different types of network and with wide array of hardware and software. Windows 10 also provides:

- Improved driver support
- Increased support for new generation hardware multimedia technologies.

4. More powerful For all your computing needs Windows 8provides:

- Industrial-strength reliability.
- The highest level of security
- Powerful performance.
- Kernel features

The kernel is considered to be the heart of the operating system that provides services to the programs running on the computer. It takes care of the hardware, software, network resources, file systems and the remaining services such as

- Security
- System fault tolerance
- Multitasking
- Multiprocessing
- Platform independence
- File system reliability
- File system security
- Flexible protocol support
- Support multi-client operating system
- Enhanced scalability
- Multi-user environment
- Communication.

4.3 Module Description

4.4.1 Analyse module

Images are breaked down to sections and are ranked using tags, then the section with most tags as an object is deemed as an object. Tracking is achieved by associating target objects in sequential frames of a video. This association can be very hard to accomplish when the objects are moving fast in relation to the frame rate of the video. Things get even more complicated when tracked objects change their orientation over time. In this scenario, video tracking systems normally use a motion model which details how the image of the target might look for several possible orientations of the object. The centroid tracking algorithm assumes that we are passing in a set of bounding box *(x, y)*-coordinates for each detected object in ***every single frame,*** Once we have the bounding box coordinates we must compute the “centroid”, or more simply, *the center (x, y)-coordinates* of the bounding box., unique identities are assigned it is done for each frame of the video, then we measure Euclidean distance between centroids, then closest pairs are deemed to be of same unique id, **The primary assumption of the centroid tracking algorithm is that a given object will potentially move in between subsequent frames, but the *distance* between the centroids for frames and will be *smaller* than all other distances between objects.** then we register new objects and de register old objects

4.4.2 Counting

Vehicles are counted when they leave the frame or cross a line at an exit point of the frame. Using a counting line makes it easier to count vehicles moving in a certain direction.

4.4.3 Plotting

When the script exits the loop, the result file is opened and the result list element is appended

CHAPTER 5

SYSTEM DESIGN

5.1 Fundamental design concepts

A set of fundamental design concepts have evolved over the past decade and each has stood the test of time. Each one provides the software designer with a foundation from which more sophisticated design methods can be applied.

5.1.1 Abstraction

Abstraction means to concentrate on a problem at some level of generalization without regard to irrelevant low-level details, user abstraction also permits one to work with concepts and terms familiar in the problem environment without having to transform them to an unfamiliar structure. Two types of abstraction are, procedural and data abstraction. Procedure abstraction is a named sequence of instructions that has a specific limited function. Data abstraction is a named collection of data that describes a data object.

5.1.2 Modularity

Modularity is an attribute that allows a program to be intellectually manageable.

5.1.3 Software Architecture

Software architecture embodies modularity, that is, software is divided into named and addressable component called modules that are integrated to satisfy problem requirements. It is the overall structure of the software and the ways in which that structure provides conceptual integrity. Program structure is the organization of control.

5.1.4 Structural Partitioning

The program structure should be partitioned both horizontally and vertically. Horizontal partitioning defines separate branches of modular hierarchy for each major problem function, vertical partitioning called factoring means control and work be distributed top down in program architecture. Top level modules should do the controlling while the lower ones should do the input, output and processing.

5.1.5 Data Structure

It is a representation of logical relationship among individual elements of data. It effects the final procedural design. It dictates the organization, methods of access, degree of associatively and processing alternatives for information. The organization and complexity of a data structure are limited only by ingenuity of the designer. E.g.: Linked List.

5.1.6 Software procedure

It focuses on processing details of each module individually. Procedure must provide specific processing details like sequence of events, exact decision points, repetitive operations and even data organization.

5.2 Input design

Input design is the link between information system and the uses and those steps necessary to put transaction data into a usable form for processing data entry. The activity of data entry can be done by instructing computer to read from a printed document or by keying data directly.

Input design focuses on controlling the amount of input required, error control, delay avoidance, simplicity. This is done by the system analyst.

Some of the considerations of Input design are:

- What data to input?
- What Medium to use?
- How the data is arranged and coded?
- The dialogue to guide the users providing input.
- Data items and transactions needing validation to detect errors.
- Methods to perform input validation and steps to follow when error occurs.

5.3 Output design

Designing output should proceed in well thought manner. The term output means any information produced by the information system whether printed or displayed. When analysts design the output, they identify the specific output that is needed to meet the requirements. The output will carry according to the type of user. Efficient and intelligent output design should improve the system relationship with the user and help in decision making. Since reports are used by management for decision making the output must be simple, descriptive and clear to the user. Options for outputs are given through system menus.

Some of the considerations of output design are:

- Determine the information to present
- Decide whether to display, print the information and select the output medium.
- Arrange the information in acceptable format.
- Decide how to distribute the output of intended recipient.

5.4 Development Approach

The development approach is based on the incremental model. This model emphasizes ability to keep improving the current version. Testing software is an activity following the implementation phase. In each phase of the development process comparison of results is done against what is required. Quality has to be also assessed and controlled

5.5 Data flow diagram

It is a graphical representation of the System's data and how the processes transform the data. It depicts information flow and transformations on the data as data moves from the input to the output. It is the starting point of design phase that functionally decomposes the requirement specification down to the lowest part of details. Thus, DFD describes what data flows rather than how they are processed.

To construct a DFD we use

- Arrows
- Ovals
- Rectangles

Arrows are for data flow indication. Ovals are for processes. Rectangles are for source or destination of data.

Rules for drawing DFD are:

- Arrows should not cross each other.
- Rectangles and Ovals must bear names.
- Name must be meaningful.
- Decomposed ovals must have unique names.

LEVEL 0 CONTEXT DIAGRAM

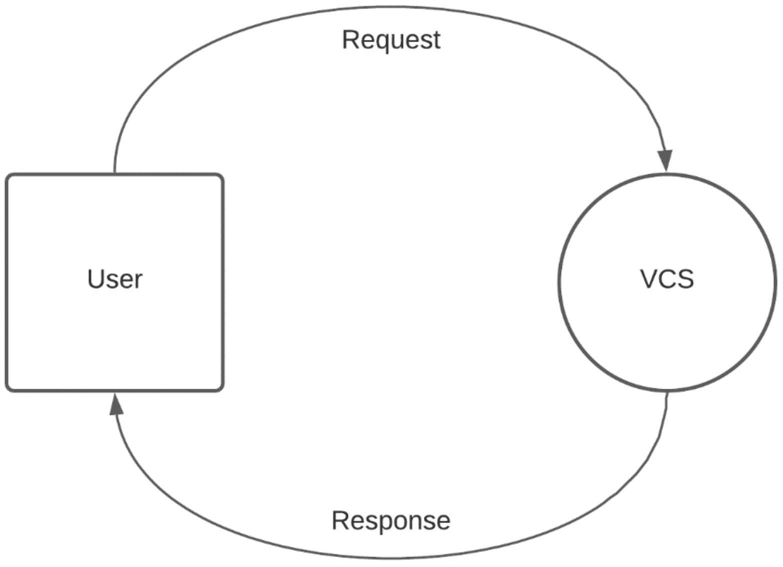


Fig 5.5.1

LEVEL 1 DATAFLOW DIAGRAM

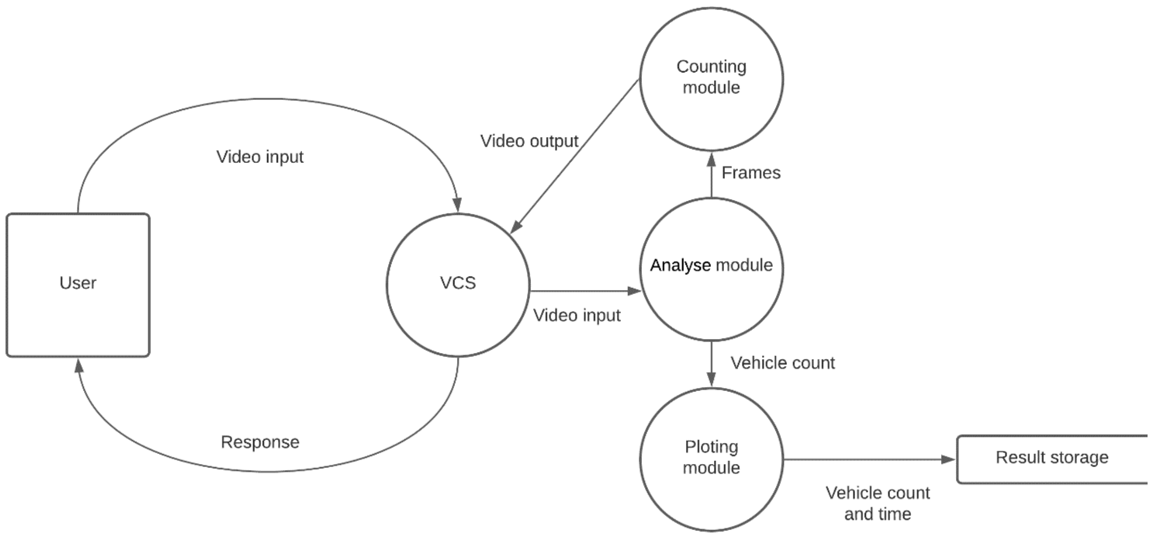


Fig 5.5.2

CHAPTER 6

SYSTEM TESTING

Testing is important in quality measure employed during software development. After the coding phase, the programs are available for testing. Testing uncovers errors because of coding, locates errors because of previous phases of development.

6.1 Testing objectives

- To ensure that during operation the system will perform as per specification.
- To make sure that system meets the user requirements during operation
- To make sure that during the operation, incorrect input, processing and output will be detected
- To see that when correct inputs are fed to the system the outputs are correct
- To verify that the controls incorporated in the same system as intended
- Testing is a process of executing a program with the intent of finding an error
- A good test case is one that has a high probability of finding an as yet undiscovered error

The software developed has been tested successfully using the following testing strategies and any errors that are encountered are corrected and again the part of the program or the procedure or function is put to testing until all the errors are removed. A successful test is one that uncovers an as yet undiscovered error.

6.2 Types of Testing

This phase uncovers bugs in the program for correction. One goal of dynamic testing is to produce a test suite. This is applied to ensure that modification of the program does not have any side effects thus improving reliability. This is called REGRESSION testing.

Types of testing are:

6.2.1 Unit testing

This is the first level of testing. Different modules are tested against specification produced during the design of modules. Unit testing helps in verification of code produced by isolating modules of each other.

After coding, each module is tested and run individually. All unnecessary coding is removed and all modules are ensured to work properly as the programmer would expect. Logical errors found are corrected. So, by working all modules independently and by verifying the outputs, the program was functioning as required.

6.2.2 Integration testing

Data can be lost while using the interface, one module can have adverse effect on another sub functions, when combined May or may not produce desired functionality. It is a systematic testing for constructing a program structure, while at the same time, conducting test to uncover errors associated within the interface. Correction is difficult because, testing of entire program complicates isolation of costs. Thus, all errors uncovered are corrected for next stage of testing.

6.2.3 Validation testing

This provides final assurance that the software meets all the functional, behavioural and performance requirements. The software is completely assembled as a package. Validation succeeds when software functions in a manner in which the user expects and is the process of using in a live environment to find errors. During the course of validating the system, failures may occur and sometimes the coding has to be changed according to the environment. Once this is done, input of dummy data ensures that the software satisfies all the requirements.

6.2.4 Output testing

After performing the validation testing, the next step is the output testing of the proposed system since no system could be useful if it does not produce the required output in either or both forms i.e. on screen or printing. The output format for onscreen was found correct as per system design.

6.2.5 User Acceptance testing

It is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system Users at the time of development and making changes required.

Preparation of test data plays a vital role in system testing. After preparing the test data, the system understudy is tested using the test data. While testing the system by using the test data, errors are again uncovered, corrected and noted for the future.

6.3 Test Case

TEST	USECASE STEP	ACTION PERFORMED	EXPECTED RESULT	ACTUAL RESULT	DO EXPECTED AND ACTUAL RESULT CORRESPOND
1	ANALYSE	READ VIDEO	VIDEO READ	AS EXPECTED	YES
2	ANALYSE	EXTRACT FRAMES	EACH FRAMES EXTRACTED	AS EXPECTED	YES
3	ANALYSE	CREATE FOREGROUND MASK	FGMASK CREATED	AS EXPECTED	YES
4	ANALYSE	CENTROID CREATION	CENTROID CREATED	AS EXPECTED	YES
5	ANALYSE	CENTROID TRACKING	CENTROID BEING TRACKED	AS EXPECTED	YES
6	COUNTER	CHECK CROSSING REFERENCE LINE	LINE CROSSING IS DETECTED	AS EXPECTED	YES
7	COUNTER	LIST UPDATING	LIST UPDATED	AS EXPECTED	YES
8	COUNTER	VIDEO OUT	OUTPUT VIDEO GENERATED	AS EXPECTED	YES
9	PLOTTER	FILE CREATION	FILE CREATED	AS EXPECTED	YES
10	PLOTTER	APPENDING DATA	DATA APPENDED	AS EXPECTED	YES

Table 6.3

6.4 QUALITY ASSURANCE

It consists of auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary about the product quality meeting tis goals.

This is applied during the engineering process.

Quality assurance consists of:

- Analysis, Design, Coding and Testing methods and tools
- Formal technical reviews that are applied during each software engineering.
- Multi-tiered testing strategy.
- Control of software documentation and the change made to it.
- A procedure to ensure compliance with the software development standards.
- Measurement and reporting measurements.

CHAPTER 7

SYSTEM IMPLEMENTATION

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage, the main workload, the greatest upheaval and major impact on existing system shift to user department. If the implementation is not carefully planned and controlled, it can causer chaos and confusions.

It includes all activities that convert from the old system to the new system which may be a totally new system. Proper implementation is essential to provide a reliable system, to meet the organization requirements. Successful implementation may6 not guarantee improvement in the organization using the new system, but improper installation will prevent it. The implementation stage involves following tasks,

- Careful planning
- Investigation of system and constraints
- Design of methods to achieve the changeover.
- Training of the staff in changeover phase.
- Evaluation of change over method.
- The method of implementation and the time scale to be adopted are out initially. Next the system is tested properly and at the same time, the users are trained in the new procedures.

7.1 Implementation Procedures

Implementation of software, refers to the final installation of the package in its real environment, to the satisfaction of the intended users, and the operation of the system. In many organization, someone who will not be operating it will commission the software development project. The people who are meant to use it have doubts, whether the software will make their job easier. So, the active user must be aware of the benefits of suing the system. Their confidence in the software must be built up. Proper guidance is imparted to user so that she/he is comfortable in using the application.

7.2 User training

To achieve the objectives and benefits expected from a computer-based system, it is essential for the people who will use it to be confident in them sue of it. As system become more complex, the need for education and training is more and more important. Education is complementary to training because, it explains the background of the resources that are used. The education sections should encourage participation from all staffs, with protection for individuals for group criticism. Education should start before any development work to enable users to maintain or regain the ability to participate in the development for their system.

7.3 Application training

After basic training on computer awareness, users will have to trained in the new application software. This will give the underlying philosophy of use of the new system, such as screen flow, screen design, type of help on screen, error types while data entry, corresponding validation check at each entry, ways to correct data entry errors. It should cover information needed by specific user/group to use this system or part of the system while imparting the training of t5he program in the application. This tari8bning may be different across different user group and across different levels of hierarchy.

7.4 Operational documentation

Once the implementation plan in decided, it is essential that the user is made familiar comfortable with the environment, education involves right atmosphere, motivating the user. Documentation provides, the whole operation of the system, being developed. Such that the user can work in a consistent way. System being developed is user friendly through application tips being displayed.

Users have to be aware what can be achieved with the new system.

CHAPTER 8

SYSTEM MAINTENANCE

8.1 System maintenance

The maintenance phase of the software cycle is the time in which a software product performs useful work. The need for system maintenance is for it to be adaptable to changes in the system environment. There may be social, technical and other environmental changes which affect a system while being implemented. Software enhancements may involve providing a new functional capability, improving user displays and mode of interaction, upgrading performance characteristics of the system. So only through proper system maintenance, will the system be adaptable.

In every Library, maintenance of library material involves continuous monitoring of the books. The material has to be dusted and cleaned at periodic intervals. The damaged and torn books have to be bound. The old books which are no longer in use have to be withdrawn from the stack. This also include physical care of the books, that is, their protection from sunlight, dust, insect, moisture and heat. The maintenance work is related to many sections of the library. The processing section makes available new material all the times and there is need to shelving these books within the already existing collection.

Maintenance work which consist of:

- Shelving and re-shelving
- Keeping books and material in order and maintaining cleanliness in the shelves.
- Security of library material.
- Library binding.

CHAPTER 9

CONCLUSION

9.1 Summary

All the requirements and goals are achieved in the project which make it simple and user friendly python script for analysing traffic cam footage and generating relevant data from it. It will help to increase the productivity of those who try to analyse traffic cam footage, as they don't have to manually look for, and record each and every instance of the object coming and going through the screen and getting confused at that. The result file generated can be used for future references to analyse data from different sources and make decisions accordingly.

"VCS – Vehicle Counting System" is a python script developed to record the number of vehicles passing through the frame to a text file and later use for further research In to the topic. It provides a relatively more accurate, easy to access & timely data output and a real-time video output.

I have completed my project without any problems, I have done this project and this system is designed using python and open source python libraries only. This project has been developed by considering all the needs given un the project successfully and the time spent in it was highly enlightening. The project was divided into modules and each module were designed, developed and tested effectively.

9.2 Future Scope

Of course there are many future scopes to his project some are mentioned below,

- Make use of machine learning object detection algorithms like YOLO, TensorFlow
- Design better UI
- Hosting the script on cloud to access from anywhere
- Adding live camera monitoring support

Chapter 10

Screenshots

10.1 Initial window

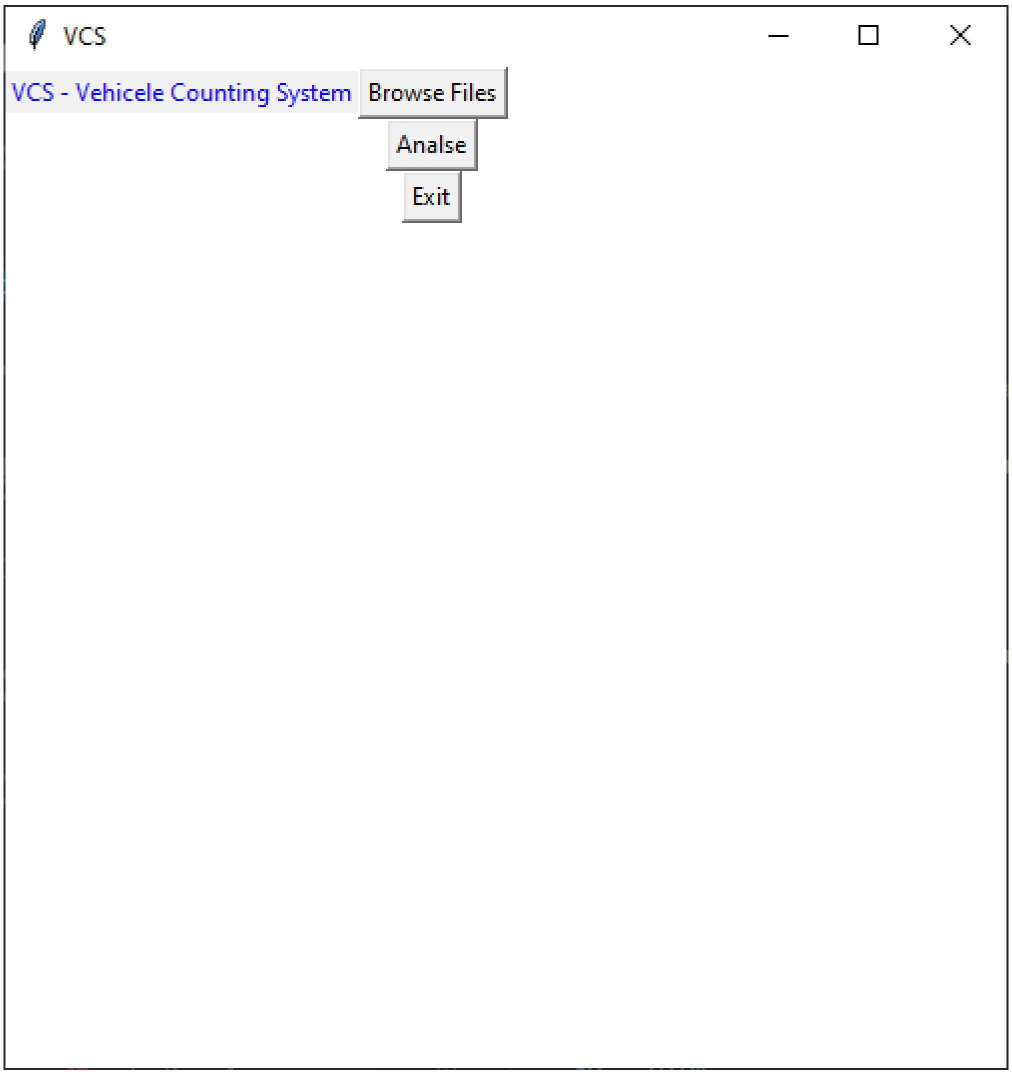
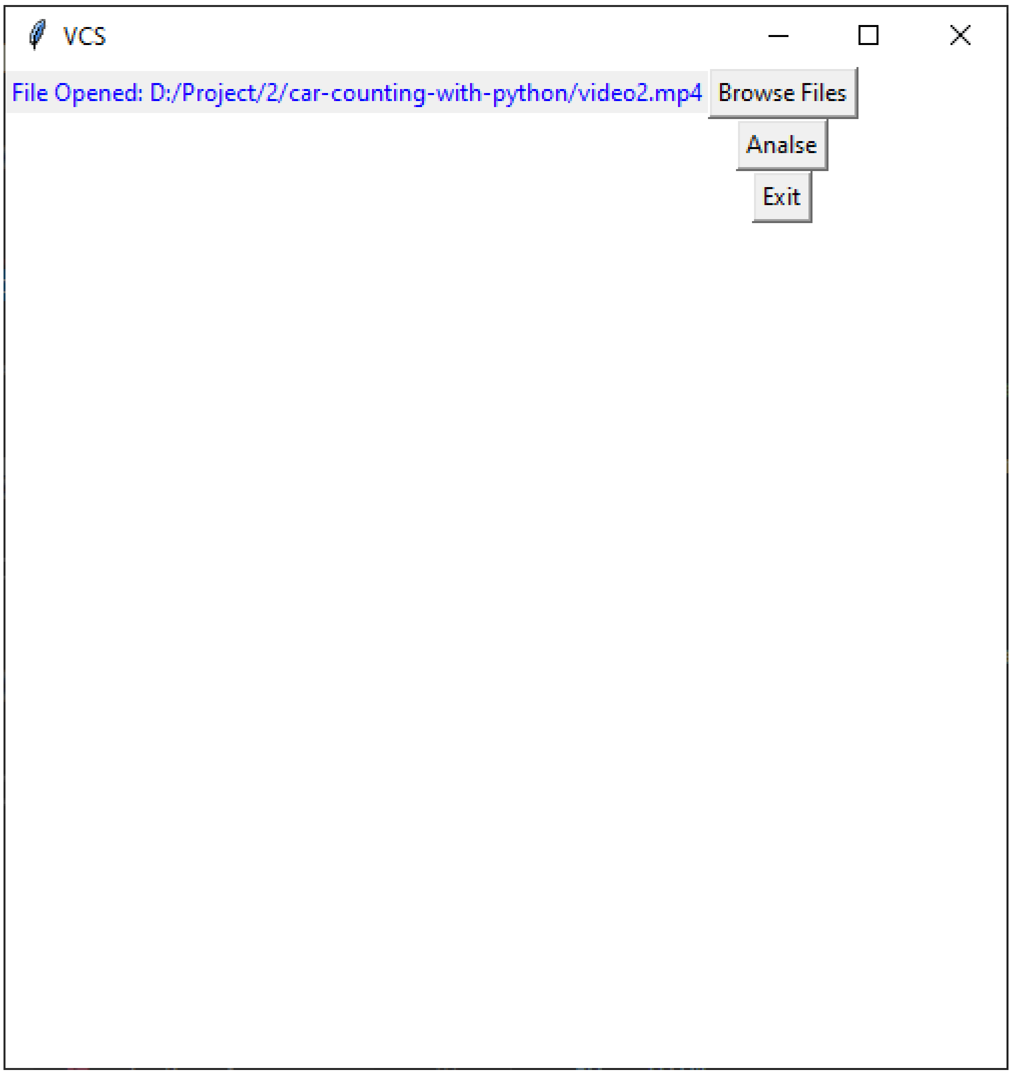


Fig 10.1

10.2 File selected



File Opened: D:/Project/2/car-counting-with-python/video2.mp4

Fig 10.2

10.3 Video output

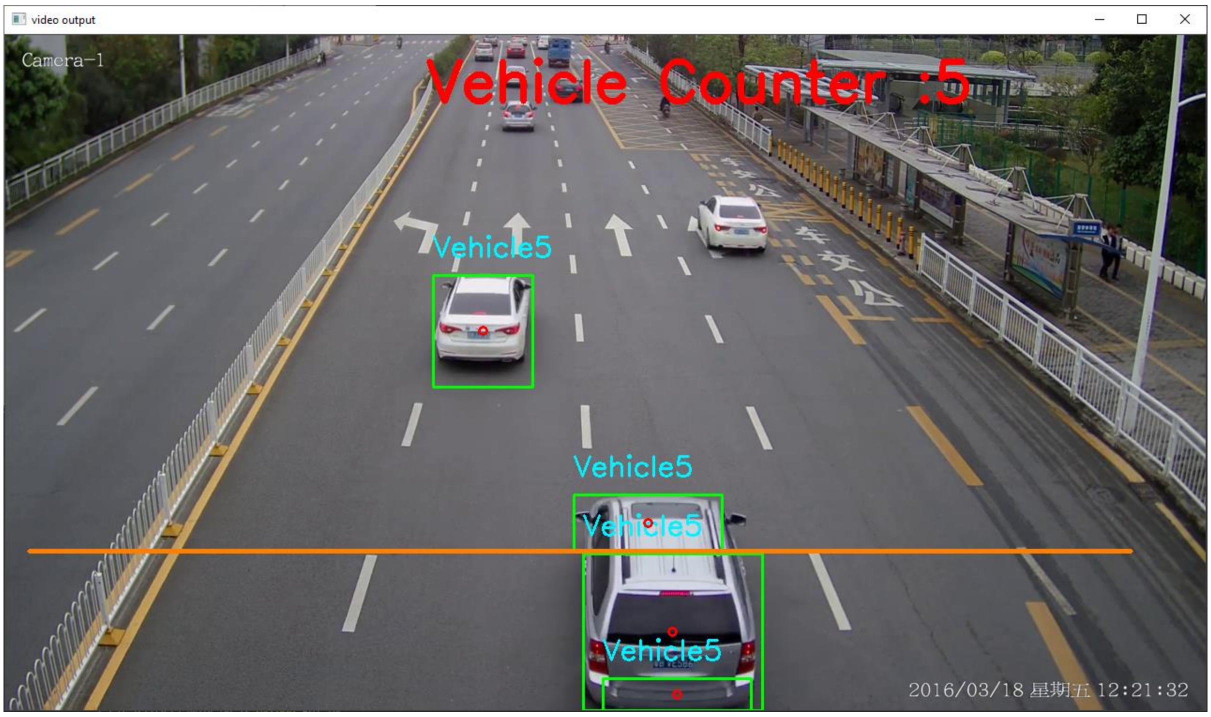
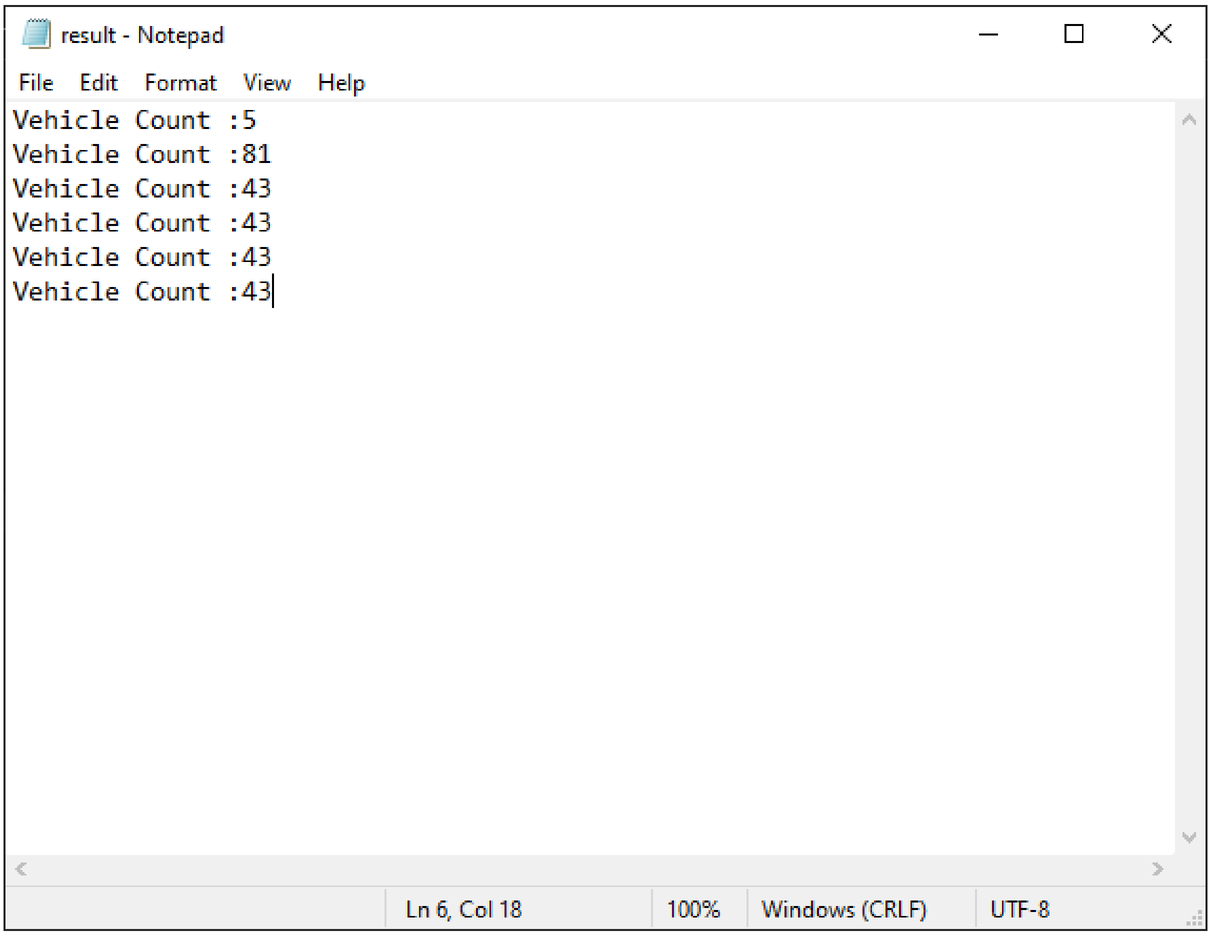


Fig 10.3

10.4 Result file



```
Vehicle Count :5
Vehicle Count :81
Vehicle Count :43
Vehicle Count :43
Vehicle Count :43
Vehicle Count :43
```

Fig 10.4

CHAPTER 11

APPENDICES

11.1 Source Code

vcs.py

```
from tkinter import *

from tkinter import filedialog

import cv2

import time

lst= []

lst2= []

count_line_pos = 550

def getlst():

    return lst[0]

# Function for opening the file explorer window

def browseFiles():

    filename = filedialog.askopenfilename(  initialdir = "/",

                                          title = "Select a video to analyse",

                                          filetypes = (("video files","*.mp4*"),("all files","*.*")))

    lst.append(filename)

    getpath()

    label_file_explorer.configure(text="File Opened: "+getlst())



def getpath():
    d = []

    d=lst[0].split("/")

    lst2.append("\\".join(d))


def centre_handle(x,y,w, h):

    x1 = int(w/2)

    y1 = int(h/2)

    cx = x+x1

    cy = y+y1

    return cx, cy


def counter(frame, count):

    cv2.putText(frame, "Vehicle Counter :"+ str(count), (450, 70), cv2.FONT_HERSHEY_SIMPLEX, 2, (0,
0,255), 5)

    cv2.line(frame, (25, count_line_pos), (1200, count_line_pos), (0,127,255), 3)

    cv2.imshow('video output', frame)


def plotter(count):

    print("Vehicle Counter :" + str(count))

    h= open("result.txt", "a")

    h.write("\n"+"Vehicle Count :" + str(count))


# analyse function

def analyse():
```

```
vid = cv2.VideoCapture(lst2[0])


ret, fr1 = vid.read()




detect = []

offset = 6

cnt = 0


while vid.isOpened():

   ret, fr2 = vid.read()

   if not ret:

      break

   frm = fr2.copy()

   msk = cv2.absdiff(fr1, fr2)


   msk = cv2.cvtColor(msk, cv2.COLOR_BGR2GRAY)


   _, thresh = cv2.threshold(msk, 25, 255, cv2.THRESH_BINARY)


   fr1 = fr2
```

```
    conts, _ = cv2.findContours(thresh, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
    for c in conts:

        if cv2.contourArea(c) < 3400:
            continue

        x, y, w, h = cv2.boundingRect(c)

        cv2.rectangle(frm, (x,y), (x+w,y+h), (0,255,0), 2)
        cv2.putText(frm, "Vehicle"+ str(cnt), (x, y-20), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,244,0), 2)
        centre = centre_handle(x,y,w,h)
        detect.append(centre)
        cv2.circle(frm, centre, 4, (0,0,255), 2, -1)
        for (x, y) in detect:
            if y<(count_line_pos+ offset) and y>(count_line_pos- offset):
                cnt+=1
                cv2.line(frm, (25, count_line_pos), (1200, count_line_pos), (0,127,255), 3)
                detect.remove((x,y))
    counter(frm, cnt)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
plotter(cnt)
cv2.destroyAllWindows()
```

```
    vid.release()


# Create the root window

window = Tk()


# Set window title

window.title('VCS')


# Set window size

window.geometry("500x500")


#Set window background color

window.config(background = "white")


# Create a File Explorer label

label_file_explorer = Label(window,

                        text = "VCS - Vehicele Counting System",

                        fg = "blue")



button_explore = Button(window,

                    text = "Browse Files",

                    command = browseFiles)
```

```
button_analyse = Button(window, text="Analse",command= analyse)


button_exit = Button(window,

                text = "Exit",

                command = exit)


# Grid method is chosen for placing

# the widgets at respective positions

# in a table like structure by

# specifying rows and columns

label_file_explorer.grid(column = 1, row = 1)


button_explore.grid(column = 2, row = 1)


button_analyse.grid(column= 2, row= 2)


button_exit.grid(column = 2,row = 3)


# Let the window wait for any events

window.mainloop()
```

REFERENCES

https://docs.opencv.org/master/

https://alphacoder.xyz/vehicle-counting/

https://seaborn.pydata.org/

https://www.w3schools.com/python/

https://www.geeksforgeeks.org/python-features/