MULUNGUSHI UNIVERSITY

SCHOOL OF SCIENCE, ENGINEERING, AND TECHNOLOGY

(SSET)

COMPUTER SCIENCE AND IT DEPARTMENT

PROJECT REPORT TITLE: DENSITY BASED TRAFFIC CONTROL SYSTEM

NAME: MAZUBA MALAMBO

STUDENT ID: 201701671

COURSE: ICT432 CAPSTONE PROJECT

PROGRAMME: BScCSIV

SUPERVISOR: MR. L. SIMUKONDA

THIS REPORT IS SUBMITTED IN PARTIAL FULFILMENT FOR THE AWARD OF BACHELOR OF COMPUTER SCIENCE FOR THE 2020/2021 ACADEMIC YEAR

DECLARATION

I, Mazuba Malambo the undersigned here, declare that the Density Based Traffic Control System is my work and to that end, this work has not been submitted to any other University, be it Local or International, to attain a degree or for an examination. In instances where I have used or quoted external material, these sources have been accredited and indicated in the complete reference list. I fully understand that the use of external material without giving due credit to the source or author may and will be considered plagiarism.

Author Name: Mazuba Malambo

Date: ……………………………………

Signature: ………………………………

Supervisor Name: Mr. L. Simukonda

Date: …………………………….

Signature: ………………………

DEDICATION

I would like to thank my supervisor (Mr. L. Simukonda) for the knowledge patience and guidance conveyed to me through the conception of the project and the time taken to help me bring my document to completion.

To my parents, Mr. Marianos M. Malambo and Mrs. Patricia K. Masule-Malambo, for all the love, care, financial, mental, physical, and spiritual support, and dedication rendered to me in my upbringing and in supporting and encouraging me to peruse greatness. Thank you both for giving me the strength to reach for the stars and pursue my dreams. There are no words to express how grateful I am to you both.

To all my friends, thank you for your motivation, understanding, and encouragement in my many, many moments of crisis. Your invaluable friendship makes my life a wonderful experience.

Above all, I thank God Almighty for the charitable time, strength, inspiration, and always being there for me.

ACKNOWLEDGEMENT

I would like to acknowledge my dissertation Supervisor Mr. Simukonda, who has been an invaluable mentor. His enduring encouragement and practical advice have been an inestimable source of support for me during this process. I would also like to thank all lecturers in the School of Science Engineering and Technology for the excellent support and Excellent knowledge delivery that they have rendered to me.

ABSTRACT

Road traffic congestion at road intersections is a growing issue for which road traffic users contend daily. The standard traffic light uses a fixed time of allocating equal green-time to lanes of traffic at road intersections regardless of the density of traffic on each lane. This is a serious problem in most cities and therefore an automated system with decision-making capabilities would work best to help replace the manual mode of a fixed timer. The current traffic signaling system is based on a predetermined period, which might be wasteful when one lane is open while the others are not. In order to solve this issue, a density-based traffic control system can be implemented. Higher traffic congestion on one side of the intersection may necessitate a longer green period than the regular permitted period. Therefore, a system where the length of green light and red light is assigned based on the density of the traffic present at the intersection at that point in time can be implemented.

The designed system uses cameras, which will detect the traffic density via image processing and change the junction timing automatically to accommodate the movement of vehicles smoothly avoiding unnecessary waiting time at the junction. The density of the vehicles is measured in three zones i.e., low, medium, high based on which zone is detected, the timings are allotted accordingly. Once the density is of a particular lane is calculated, the green time is then assigned accordingly. The cameras which are present on the road will capture the image of the lane and send the information to the system where it will decide how long the required green time is supposed to be.

Keywords: Traffic Density, Traffic Light, '+' Road Intersection.

LIST OF ACRONYMS

COCOMO	Constructive Costing Model
DBTCS	Density Based Traffic Control System
GUI	Graphical User Interface
MSCOCO	Microsoft Common Objects in Context
OpenCV	Open-Source Computer Vision Library
SDLC	Software Development Life Cycle
YOLO	You Only Look Once

LIST OF FIGURES

Table of Contents

CHAPTER 1 - INTRODUCTION

1.1 Introduction

The continuous migration of people from rural to urban areas in search of a better world has brought about population growth in cities and overburdened infrastructure. An example of an overburdened infrastructure is the road, a condition that has resulted in increased traffic. Even though traffic lights have mostly been used for controlling the flow of traffic, the management of traffic in urban cities around the world has been a source of worry. Quick transportation systems inclusive of rapid transit systems are the nerves of economic development.

Traffic mismanagement, as well as traffic congestion, results in long waiting times, loss of fuel, and money. Traffic congestion or traffic snarl-up is now considered to be one of the largest problems in urban settlements (Udoakah & Okure, 2018). Traffic problems are widely increasing as an expected result of the growing number of transportations means and the current low-quality infrastructure of the roads.

With the drastic increase in the populace of cities as well as the busyness of roads, there arises a need for a more dynamic system that seeks to reduce unnecessary congestion as well as unnecessary wait times at traffic intersections. It would be best that a new design is developed to improve how traffic flows and how it is controlled based on the density of vehicles on a particular lane by giving the right amount of time to the lane with more traffic density, hence, reducing traffic congestion in the lanes. Studies have been carried out to improve the current traffic control system which is the fixed timer because it is rendered inaccurate or inefficient when the is more traffic in one lane compared to the other (Panjwani et al., 2014).

Traffic congestion may come about due to heavy traffic at an intersection. To avoid traffic congestion, there are so many traffic management techniques available. But there is no technique that is perfect by itself as the real-time situations are constantly changing and the system has to adapt itself to change in the continuously changing circumstances.

The main aim of designing and developing the Density-based traffic control system is to reduce the waiting time of each lane of the cars and also to maximize the total number of cars that can cross an intersection. In this system, the time assigned to the traffic light of a particular lane is based on the current traffic density on the road.

1.2 Problem Statement

The present issue is that the current system of traffic lights uses a fixed allotted time and as such regardless of the traffic condition at an intersection, each lane is given equal time. The current system in place also causes unnecessary wait times at traffic intersections in an event that there is only one car and the rest of the lanes are empty. This disrupts the smooth flow of traffic and adds to the already existing issue of traffic congestion.

1.3 Aim

To develop a Density Based Traffic Control System that uses image processing to help reduce traffic congestion

1.4 Objectives

The main objectives of the system are:

- Use image processing to detect the density of traffic.
- Control traffic lights timer based on the density of traffic.
- Reduce wait times at traffic intersections

1.5 Project Scope

The scope of this project is to develop a density-based traffic control system with the use of image processing. In this project, a density-based traffic system will be developed to reduce traffic congestion at intersections. The system will be able to analyze traffic at intersections and then make a decision based on the amount of traffic present.

The value of traffic density measures only the ratio between the density of the vehicles and the total density of the road. So based on this measure, the traffic control system will compare different roads in the intersection and decide on the traffic light and the time interval given.

1.6 Project Justification

This designed system is aimed at reducing traffic congestion at intersections. The current system uses a fixed time allocating equal "GO" time to lanes of traffic at road intersections regardless of the density of traffic on each lane. The current system isn't accurate because sometimes higher traffic density at one side of the junction demands longer green time. as compared to the standard allotted time. The designed system will use a mechanism in which the period of green light and red light is assigned based on the density of the traffic present at that time and not that standard fixed allotted time.

The system will be very useful because it will also eliminate the unnecessary wait time that occurs when there is only one car at an intersection that is forced to wait for a green light instead of just proceeding because there is no impeding traffic. The designed system uses image processing technology that will be used to verify the density of traffic at the intersection.

1.7 Conclusion

The contribution presented in this designed system will help combat traffic congestion at road intersections. It will employ the use of image processing which will be more effective at controlling the flow of traffic as compared to the standard fixed timer system. Therefore, this system is developed because it will help eliminate unnecessary wait times at traffic lights as well as help improve the flow of traffic.

CHAPTER 2 – LITERATURE REVIEW

2.1 Introduction

Various Traffic Systems are used around the world to try and enhance and improve the flow of traffic. These systems are somewhat similar in the sense that they have the same aim of enhancing the flow of traffic but they also differ in the mechanisms used to control traffic. This chapter simply gives a definitive review of the components of the system and a review of existing systems. It outlines the fields of study subject to the project research, then then, it reviews the systems that are used in the mentioned technologies and study areas, and thereafter it compares the reviewed systems and discusses the proposed system.

2.1.1 History of Traffic Lights

Traffic control began to appear in London, England in the early 18th century (Michels, 2014). As early as 1722, the various traffic control measures were implemented to make sure that the movement of horse-drawn carriages, carts, and pedestrians across the London Bridge was maintained or rather kept in order. During that time, crossing the bridge was a challenge or problem because of the disorderly nature of the traffic movements. To try and deal with the challenge at hand, the Lord Mayor had organized a group of three men who were appointed as public servants to monitor and help regulate individuals crossing on the bridge.

Control of traffic in the United Kingdom began in the 1860s when the city's Police Department was tasked with managing the reckless driving of buses and carriages drawn by horses within the city. The reasoning behind the decision was the public outcry of the death of several pedestrians at the hand of horse-drawn carriages. The tallest officers on the force of the police department were appointed to the new squad to ensure that they could see over carriages and pedestrians. They were instructed to point, wave, and shout to control traffic on the busy streets of the city (Michels, 2014).

2.2 Review of Existing Systems

2.2.1 Manual Traffic Control Managements

Manual traffic control is a common intersection control technique in which trained personnel, law enforcement officers, give the right-of-way to approaching vehicles at a traffic intersection (S. Parr & Wolshon, 2015). This is the simplest form of traffic management, which mainly includes humans in the method. An example of a traffic control mechanism that is manual is a traffic policeman is standing at cross-sections of roads controlling the flow of traffic by using signs as shown in the figure below

Figure 2. 1 Traffic Officer controlling traffic flow (https://old.keralapolice.gov.in/wings/specialized-wings/traffic-police).

Another means of manual traffic control is via the use of road traffic signs and road markings. Traffic signs and road markings are used to control and provide a smooth and orderly flow of traffic. Apart from guiding traffic, Traffic Road signs and road markings also warn drivers. Below are images of some road signs and markings.

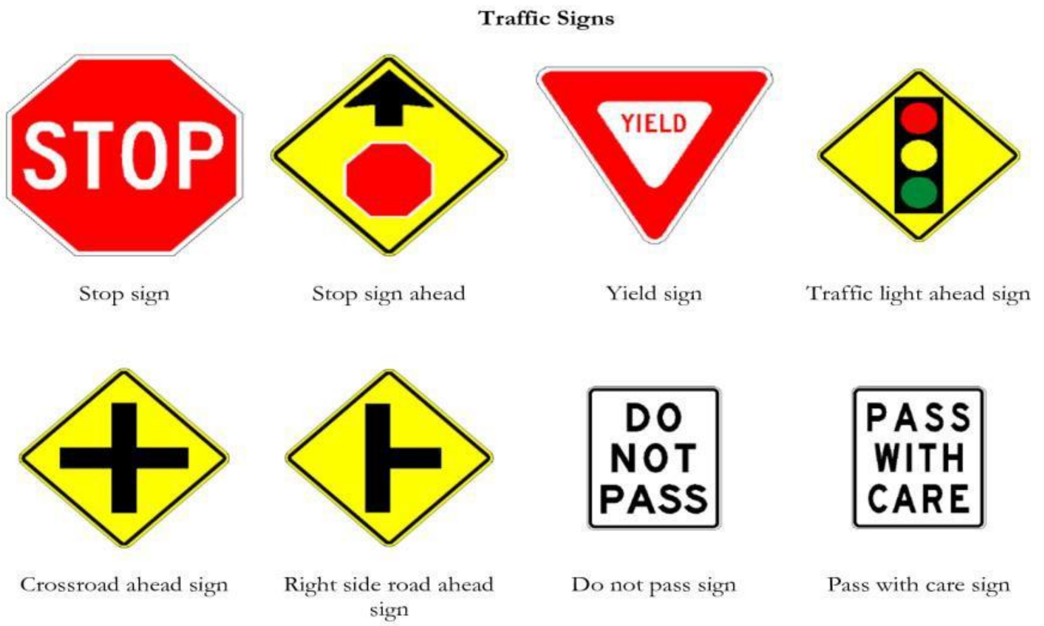


Figure 2. 2 Examples of road signs (Slinn et al., 2005)

Figure 2. 3 Road Markings (Slinn et al., 2005).

Traffic signals, road signs, and road markings can be combined or used individually to convey the maximum information available to a road user.

A. STRENGTHS

The advantage of using manual traffic control management is that as traffic density varies randomly, a traffic officer can allocate the best time appropriately at any given instance (S. A. Parr et al., 2020). Another is that traffic officers can use common sense judgments in varying situations and pedestrians can cross the road safely.

B. LIMITATIONS

Having an officer control traffic is much more expensive than an automated system. Traffic officers are prone to get tired thus having so many humans do the job. Lack of trained traffic police officers makes this problem worse in many cities. In bad weather conditions, it is next to impossible to be able to manually manage traffic successfully. Below is an image depicting traffic control in bad weather.

Figure 2. 4 Traffic Officer controlling traffic in bad weather. (https://zambiareports.com/2018/05/17/police-officer-rainstraffic-control/cop-rains/)

If the weather gets to an extreme condition, the traffic officer is at risk of getting an injury thus it being a disadvantage.

2.2.2 Automatic Traffic Management Technique

Stoplights, sometimes known as robots, are traffic-controlling devices that are placed at road junctions, pedestrian crossings, and other sites with the aim to maintain and control the traffic. As a way of trying to eliminate most weaknesses of the manual traffic control system, an automatic traffic management method is implemented (S. Parr & Wolshon, 2015). This system includes a simple three-color traffic signal which is Red: Tell traffic to stop or prohibits any traffic from proceeding, Amber or Yellow: This simply warns the road user that the light is about to switch to red thus requiring drivers to come to a slow down stop or the signal is about to change to green allowing drivers to get ready to proceed and Green: Allows traffic the traffic flow to proceed in the direction that is intended. Below is an image depicting a traffic light.

Figure 2. 5 Traffic Light

A traffic light is composed of the main display which has red, orange, and green lights. The lights on the main display are made of Led lights which stand a test of time before they require replacement or servicing. Traffic lights also have a signal cabinet or control box where the traffic controller is stored. The signal cabinet houses a power panel in charge of electrical distribution, the controller itself which is the brain of the whole system. And lastly supports such as the body of the traffic light systems.

A. STRENGTHS

Traffic lights provide authority to drivers to move with confidence as they know that the light has permitted them to move. They also help or make traffic move in an orderly manner as well as reduce the number of road accidents such as pedestrian accidents. Traffic light indications are also visible in foggy weather as compared to the conventional manual system.

- *B. LIMITATIONS*

Traffic lights slow down the flow of traffic at peak hours. Also, when there is a signal breakdown there is a widespread of traffic difficulties. Rear-end collisions are prone to happen at traffic lights. Also, unjustified traffic signals can cause excessive delays, disobedience of signals, and diversion of traffic to inadequate alternate routes.

2.2.3 Traffic Control System Using Inductive Loop Detection

An Inductive loop is simply a detection system that applies the use of a moving magnet or an alternating current intending to induce an electric current in a nearby wire. They are made of wire "coiled" to form a loop that usually is a square, circle, or rectangle shape and it is installed into or under the surface of the roadway for communication signal transmission and reception so as to detect the presence of vehicles. Inductive loops work like metal detectors in the sense that they measure the change in the field when objects pass over them e.g. when a car drives over a loop sensor the loop field changes which allows the detection device to detect the presence of an object (car). Below is an image depicting an inductive loop.

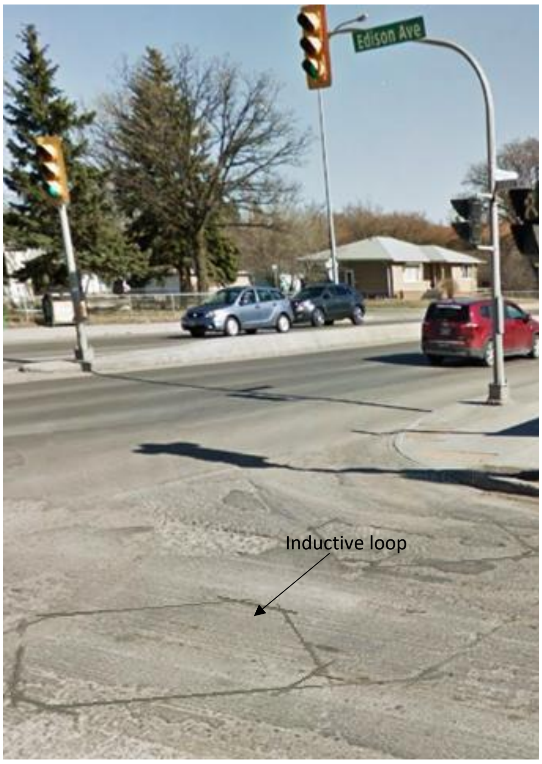


Figure 2. 6 Inductive Loop at Traffic Intersection

Bhaskar et al., (2016) proposed that the system will use two inductive loops in each lane to measure traffic density. The traffic density is to be classified into three levels namely normal, medium congested, and extremely congested. In the event that there is a red signal in LANE-1 at an intersection. The vehicles will come and stop causing the loop-1of LANE-1 to go high. This signifies that the cars that have been halted have come to a standstill up till where the first loop has been installed. As a result, traffic in this lane might be described as somewhat crowded. When both circuits of both loops of a lane are detected, the signal turns high at the same time and this indicates that there are cars that are at a standstill up until or extending beyond the second loop. If none of the loops detect the presence of a vehicle at a red light, then that lane has normal traffic. As a result, loop-1 will be used to measure medium congestion, whereas loop-2 will be used to assess high congestion. The proposed system was to only allow one lane to be functional at an intersection while the other lanes await a green light. Below is an image depicting the proposed system with a timer.

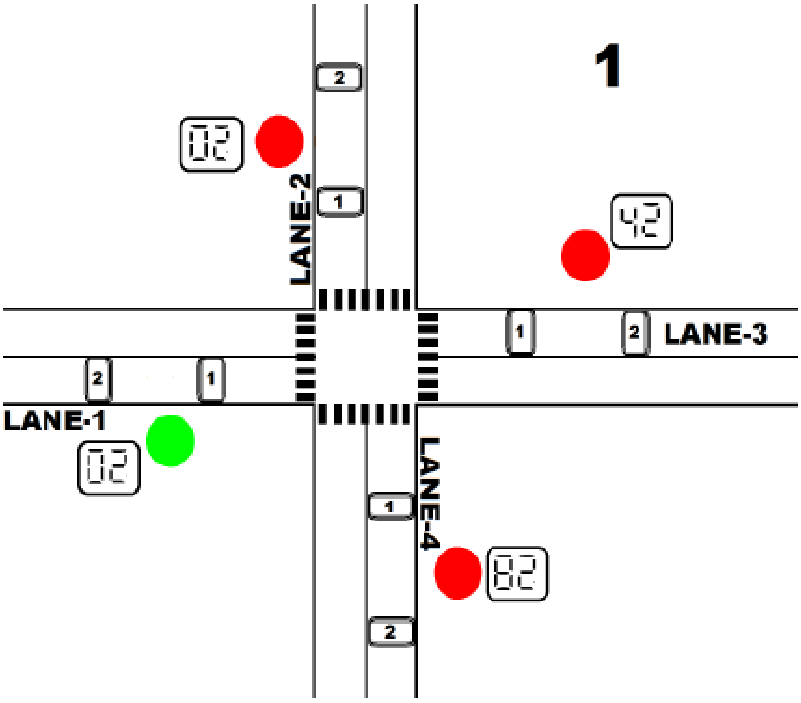


Figure 2. 7 Traffic lights with inductive loops with a timer of 40sec (Bhaskar et al., 2016).

A. STRENGTHS

Inductive loops are accurate and reliable at detecting the density of traffic. They are inexpensive and they also consume little power. They are also convenient and hidden since they are installed under the road.

B. LIMITATIONS

As much as inductive loop detections are highly accurate, they have challenges in detecting small cars. They are sensitive to temperature fluctuations and they also get damaged easily as the road begins to deteriorate because of heavy vehicle/portholes. Inductive loop detection is also affected by the metallic particles in road construction material and the is a high risk of theft of the loop and feeder cable.

2.2.4

Below is a table that shows the comparisons of the reviewed systems.

Table 2. 1 Comparison of Systems

		Fixed Time	Vehicle Detection	Real-Time Congestion Resolution	Working Time (24/7)
System	Manual Traffic Control Managements (S. Parr & Wolshon, 2015)	NO	YES	NO	NO
	Automatic Traffic Management Technique (Pashupatimath, 2018)	YES	NO	YES	YES
	Traffic Control System Using Inductive Loop Detection (V.S & Shaithya.V, 2015)	NO	NO	YES	YES
	Proposed System	NO	YES	YES	YES

2.3 Proposed System

The designed system is aimed at making the traffic signal to be more efficient and feasible. Therefore, reducing traffic as well as utilizing the time. The designed system is density-based which means it will give priority to a particular lane based on the number of vehicles. Below is a formula that show how the time will be calculated

$$\frac{Number\ of\ vehicles\ in\ particular\ lane}{Total\ No\ of\ vehicles\ in\ all\ lanes} * Time\ Delay$$

The formula above shows how the green time will be calculated for a particular lane. The time delay value is a fixed value that is used to calculate the green time for a particular lane. Based on the time delay, there can be an increase or decrease in green time.

Image processing is the method that will be used for calculating the density. Python will be used as a simulation environment.

2.4 Review of Software Development Methodologies

2.4.1 Waterfall Model

This is a sequential model that divides the SDLC into pre-defined phases that must be completed before the next phase can begin. Each phase relies on the deliverables of the previous phase (Sommerville, 2006). It has a less flexible approach as progress flows in one direction. Its phases are Requirements Analysis, System Design, Implementation, Testing, Deployment, and maintenance.

2.4.2 V-Model

This model is simply a variation of the waterfall model in a V shape. The half of the V shape represents the evolution of user requirements while the right half represents the integration and verification of the system components into successive levels of implementation (Sommerville, 2006). It is also known as the Verification and Validation model because each development activity has a corresponding testing phase.

2.4.3 Incremental Model

The incremental is simply based on the thought of developing an initial product and exposing it to the user allowing them to comment which in turn helps it evolve into several versions. In the incremental model, it's simpler to make modifications to the software when it's still in development. Each and every increment or version of the product or system contains capabilities that the user requires (Sommerville, 2006).

2.4.4 Spiral Model

In this model, the software process is depicted as a spiral instead of the normal sequence of events. Each and every loop in the spiral depicts the various phases in the software development lifecycle. Change avoidance and change tolerance are combined in the spiral model of software development. It is assumed that system modifications are the outcome of project risks. (Sommerville, 2006).

2.4.5 Prototyping Model

In Prototyping, an initial version of a product or system is used to demonstrate different concepts and try out various design schemes. It creates the base or foundation on which the final system is laid upon. While the system is being created, a system prototype may be used to do design experiments to see if a suggested design is feasible.

2.4.6 Agile Model

The agile model is simply an umbrella term for a set of various methods and practices that are based on the values and principles expressed in the Agile Manifesto that is a way of thinking that enables teams, groups, as well as businesses to innovate, quickly respond to the changing demand, while mitigating different risks.

2.5 Selected Methodology

The methodology used for this project is the waterfall model. Each phase in the Waterfall model has to be completed fully before the next phase can commence and thus, phases don't overlap. The Waterfall software development model is used for a small project where no uncertain requirements are present. After each phase is completed, a review is done to determine if the project is on the right path and whether to continue or discard the project. This model software tests the software only after the development is complete. Below is an image of the waterfall model.

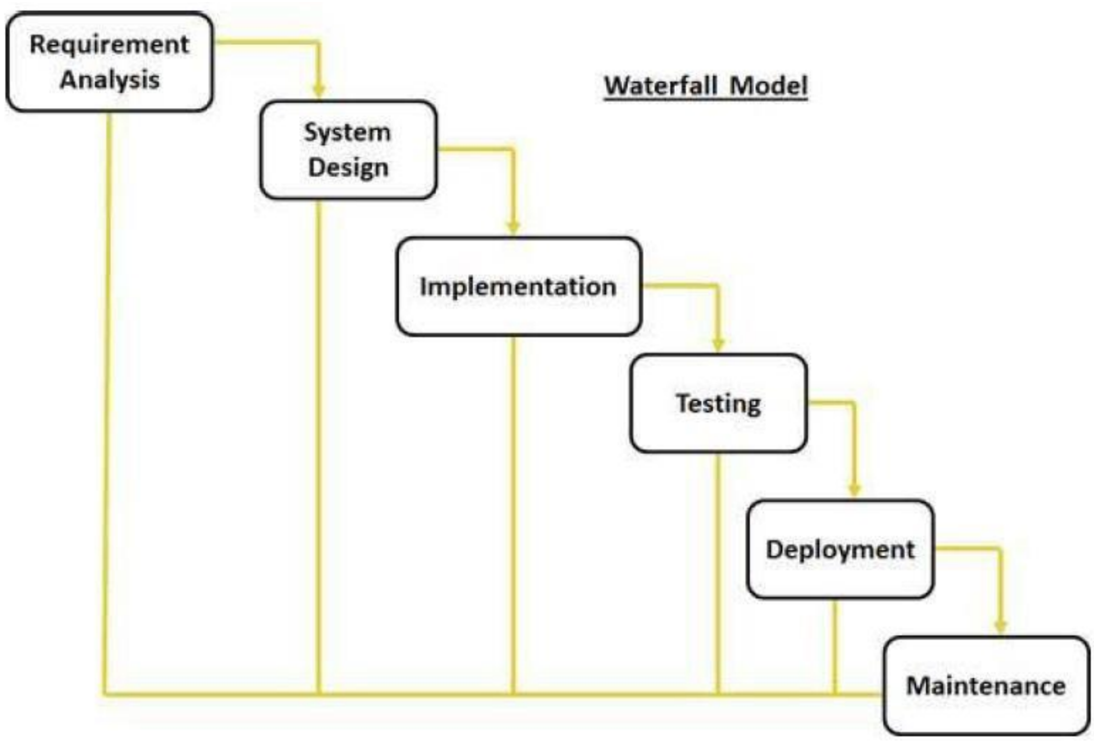


Figure 2. 8 Waterfall Model Phases (Bulman, 2017)

The phases of the waterfall model are:

2.5.1 Requirements Analysis

This is the first and initial stage of the waterfall model where all the requirements for the system to be developed are captured. Requirements are captured or collected from the end-user of the system. At this phase of the project requirements such as traffic data at intersections were gathered. Other requirements such as how the system should behave when there is traffic and when there isn't.

2.5.2 System Design

This is the second phase of the Waterfall model. Its function is to study the requirements that are listed in phase one in preparation for system design. This process allocates the required resources to either hardware or software by establishing an overall architecture. The software design process involves the identification and description of fundamental system relationships and their abstractions. During this phase, the system was designed in terms of what software and frameworks to be used such as OpenCV and Python programming language.

2.5.3 Implementation and Unit Testing.

In this phase, the coding of the system is commenced and the system is developed into programs called units. The individual program units are tested to verify that each of them meets its specification. At this stage, individual parts of the system are designed such as image retrieval, image conversion as well as the traffic timer after the density has been assessed.

2.5.4 Integration and System Testing

The various program units are integrated to form a complete system and it is tested for proper coordination among system modules as well as its behavior. Once this phase is completed, the software is delivered to the customer. At this phase, all the individual units are bundled into one whole system and it's tested for its functionality between the individual units.

2.5.5 Operation and Maintenance

Operation and Maintenance is the longest phase in the Waterfall model. The system is deployed and installed and placed in a running environment. Once put to practical use, problems arise and thus maintenance comes in. Maintenance helps to solve issues that are related to the system that arise only after deployment of the system. The system is then deployed and if problems arise, maintenance is carried out.

2.6 Technologies and framework used

2.6.1 Review of Programming Languages

A programming language is simply a formal language that is comprised of a set of strings that produce different kinds of machine code output. They contain instructions that can be used to create various types of programs and implement algorithms. Different programming languages have different properties and features that determine what type and kind of application can be developed with their use. There are various programming languages some of which are

2.6.1.1 Java

Java is a high-level, powerful general programming language that is used to design and develop desktop and mobile applications. It runs on a variety of platforms such as Mac OS, windows, and the various versions of UNIX. Java has the following properties:

- It is objected oriented and thus it can be easily extended since it is based on the object model
- It is platform-independent. This simply means when Java is compiled, it is not compiled into a platform-specific machine but rather into platform-independent byte code
- It is robust. This means it makes various efforts to try and eliminate error-prone situations by emphasizing on compile-time error checking as well as runtime checking.

2.6.1.2 Python

Python is a high-level, interpreted, and general-purpose programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming (van Rossum & Python development team, 2018). According to Nosrati, (2011) some of the characteristics of python are that:

- It is open source
- It is fast and powerful
- It is portable
- It supports other technologies

2.6.1.3 C

C programming is an imperative, general-purpose programing language developed by Dennis M. Ritchie to develop the UNIX operating system. It is a successor of the B language and it is one of the most widely used programming languages alongside java. Some of its properties are:

- It is fast and efficient
- It has a rich set of built-in operators
- It has libraries with rich functions
- It is a procedural language.

2.6.1.4 C#

This is a simple general-purpose programming language that was developed by Microsoft. It is based on C and C++ programming languages. It can be used for a variety of things, including mobile applications, game development, as well as enterprise software.

2.6.2 Review of Database Technologies

A database is simply a structured set of data or an organized/ordered collection of data that is generally stored and accessed electronically from a computer system. There are different types of databases that all have their own features and properties. Some of the different types of databases are:

2.6.2.1 Hierarchical Database

These employ the parent-child relationship of storing data. It is structured like a tree that has nodes representing records and branches representing fields. An example is the IBM information management system.

2.6.2.2 Relational Database.

These employ database relationships in the form of tables. A table consists of rows(tuples) and columns (attributes). It uses SQL to manipulate the data. An example is MySQL.

2.6.2.3 Object-Oriented Database

These support the storage of all data types and the data is stored in the form of objects. The objects have attributes and methods that define how the data is to be handled. An example is Mongo DB

2.6.2.4 Centralized Database

This employs data storage at a central site and different users from remote locations can access the information. It contains application procedures that help the different users to access the data even from a remote location. Different kinds of authentication procedures are applied for the verification and validation of the system's end users.

2.6.2.5 Personal Database

In this type of database, data is collected and stored on personal computers which are easily manageable. The data is generally used by the same department of an organization and is accessed by a small group of people.

2.6.2.6 NoSQL Database

These are normally used for large sets of distributed data. It is very efficient at analyzing a large size of unstructured data. It has a hierarchy that is similar to a file folder and that data within is unstructured. An example is Couchbase.

2.6.2.7 Cloud Database

This is a database that runs over the internet. The data is stored on a local hard drive or server but it is available online thus making it easy to access the files from any location as long as there is an internet connection.

2.6.3 Technologies to be used in Project

2.6.3.1 Python Programming Language

Python is a high-level, interpreted, and general-purpose programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming (van Rossum & Python development team, 2018). According to Nosrati, (2011) some of the characteristics of python are that:

- It is open source
- It is fast and powerful
- It is portable
- It supports other technologies

Nosrati (2011) also stipulated that python can be used to write programs types such as:

- System Programs
- Graphical User Interface (GUI)
- Numerical Programming
- Database Programming
- Components Integrity and many others

2.6.3.2 OpenCV

Open-Source Computer Vision Library commonly known as OpenCV is a library of programming functions mainly aimed at real-time computer vision. It is used for all kinds of image and video processing and analysis. It is capable of processing images and videos to identify objects, faces, or even handwriting (Scherfgen & Sc, 2015). OpenCV is a library of programming functions mainly used for image processing and some of its image processing features according to Naveenkumar, (2016) are:

- Image Filtering
- Image Transformation
- Object Tracking
- Feature Detection

2.6.3.3 YOLO

You Only Look Once commonly known as YOLO, is a family of models that are a series of end-to-end deep learning models designed for fast object detection (*How to Perform Object Detection With YOLOv3 in Keras*, n.d.). It is simply a real-time object detection algorithm that is used to identify specific objects in videos, live feeds, or images. It detects objects by splitting the input into a grid of cells and then each cell directly predicts a bounding box and object classification.

The result is a large number of candidate bounding boxes that are consolidated into a final prediction by a post-processing step. It uses the Microsoft Common Objects in Context (MSCOCO) dataset and it was trained on 80 object categories. The MSCOCO dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset and it consists of 328K images

2.6.3.4 PyQt5

This is a python binding that is implemented as a Python plug-in. It is a module that is used in building graphical user interface (GUI) apps in python due to its simplicity. It has a feature known as PyQt5 Designer which makes the development of complex GUI apps to be done in short times due to its drag and drop functionalities.

2.7 Summary

From the above, it can be concluded that Traffic signal optimization is dependant on several factors such as real-time traffic conditions, volume, congestion, and route choice. To optimize traffic signals of urban roads all these variables are to be considered and a best-fitted method to solve the problem can be formulated. Various methodologies are used in the software development process. This chapter provides a definitive review of the components of the system and the chosen methodology and why it was chosen. The selected methodology is suitable because requirements are not likely to change and they are well defined. An overview of the methodology was given as well as the technologies and frameworks used.

CHAPTER 3 – SYSTEMS ANALYSIS AND DESIGN

3.1 Introduction

This chapter focuses on the systems analysis design and process. According to (Sommerville, 1977), the systems development life cycle (SDLC) involved in system analysis and design has a fundamental model that has four phases which are planning, analysis, design, and implementation, common to the development of all information systems projects.

3.2 System Analysis

Harris et al., (2006) simply state that systems analysis and design is an approach to the development of information systems that encompasses the four phases of the systems development life cycle (SDLC). It is conducted to study a system and or its parts with the aim of identifying its objectives. It is also a problem-solving strategy that improves the system and It ensures that all of the system's components work hand-in-hand to achieve their goals. (*System Analysis and Design - Overview - Tutorialspoint*, n.d.).

3.2.1 Requirements Analysis

Requirement analysis is simply the process of defining user expectations for new software that is designed or modified in the sense that it helps to discover, analyze, refine, and scrutinize the gathered requirements to make consistent and unambiguous requirements (Gunda, 2008).

3.2.1.1 Functional Requirements

According to (Dabbagh et al., 2016), A functional requirement is a requirement that specifies a function that a system or component must be able to perform. Below are the functional requirements for the system:

- The system shall detect vehicles in a given image.
- The system shall count the number of vehicles in an image.
- The system shall calculate the required allotted time.
- The system shall change the traffic signal based on the allotted time.
- The system shall display the required traffic signal to the motor road user.
- The system shall be available 24/7

3.2.1.2 Non-Functional Requirements

A non-functional requirement is a statement of how a system must behave, it is a constraint upon the behavior of the system (Dabbagh et al., 2016). They specify the criteria that judge the operation of a system, rather than specific system behaviors. Below are the non-functional requirements for the system:

- The system should be easily maintainable
- The system should be environmentally friendly
- The system should be able to work at peak hours

3.3 System Design

The system design phase is what follows next after the requirements analysis phase. System design is the process of determining the overall system architecture—a collection of physical processing components, hardware, software, people, and their interactions—that will meet the system's key needs. (Dennis et al., 2015).

3.3.1 Use Case Diagram

A use case diagram is simply a graphical representation of a user's possible interactions with a system. It is accompanied by other types of diagrams and it is used to gather requirements of a system which also includes different internal and external influences. Below is the use case diagram of the Density Based Traffic Control System:

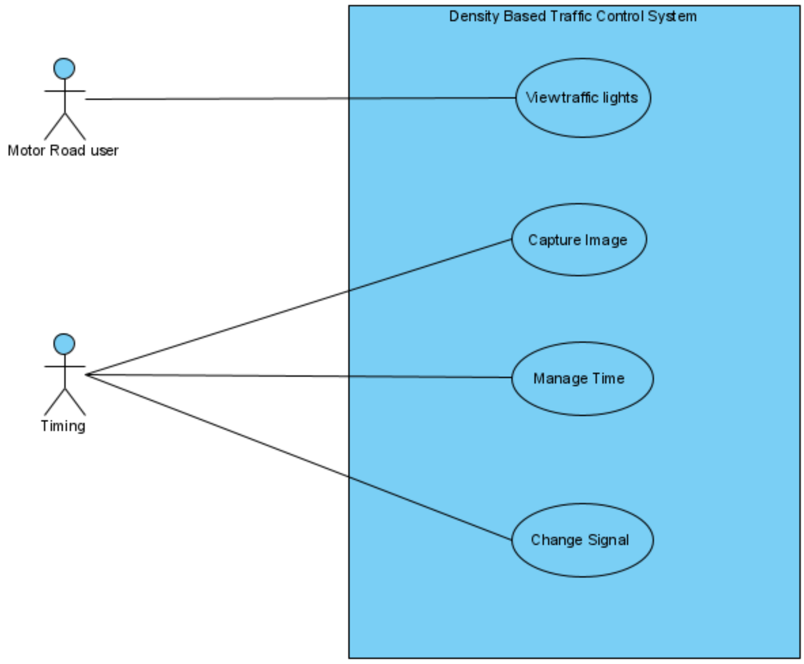


Figure 3. 1Use Case diagram for DBTCS

Table 3.1 Capture Image use case explained

Name	Capture Image	
Actor	Timing	
Description	Image of the road is captured to determine the green time to be allotted	
Action	Actor Action:	System Response:
	1. Capture image of road	2. Image is Captured
Successful Completion		
Pre-condition	None	
Post-condition	None	

Table 3.2 Manage time use case explained

Name	Manage Time	
Actor	Timing	
Description	Green time to be allotted is calculated based on traffic density	
Action	Actor Action:	System Response:
	1. Insert captured image 3. Calculate time	2. Present image with identified vehicles and count of vehicles. 4. present calculated time with traffic queue.
Successful Completion	Green time is calculated and allotted and a queue of traffic is created based on the number of cars	
Pre-condition	Image of the road must be captured	
Post-condition	None	

Table 3.3 Change Signal use case explained

Name	Change Signal	
Actor	Timing	
Description	The traffic signal is to be changed to green for a specific amount of time	
Action	Actor Action:	System Response:
	1. Insert captured image 3. Calculate time	2. Present image with identified vehicles and count of vehicles. 4. present calculated time with traffic queue.
Successful Completion	The Greenlight is awarded for a specific amount of time and traffic is allowed to move	
Pre-condition	Green time must be calculated and allotted and a queue of traffic must be created based on the number of cars	
Post-condition	None	

3.3.2 Dataflow Diagram

The diagram below shows the data flow diagram of the density-based traffic control system at Level 0. It can be seen that the captured image is sent to the system after which the green time is calculated and the signal can be changed.

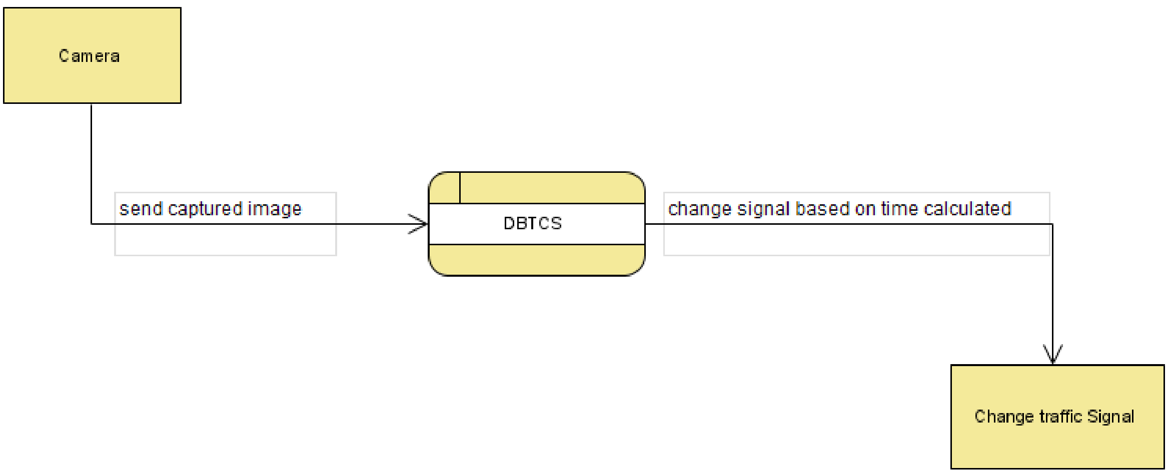


Figure 3. 2 Level 0 Dataflow diagram for DBTCS

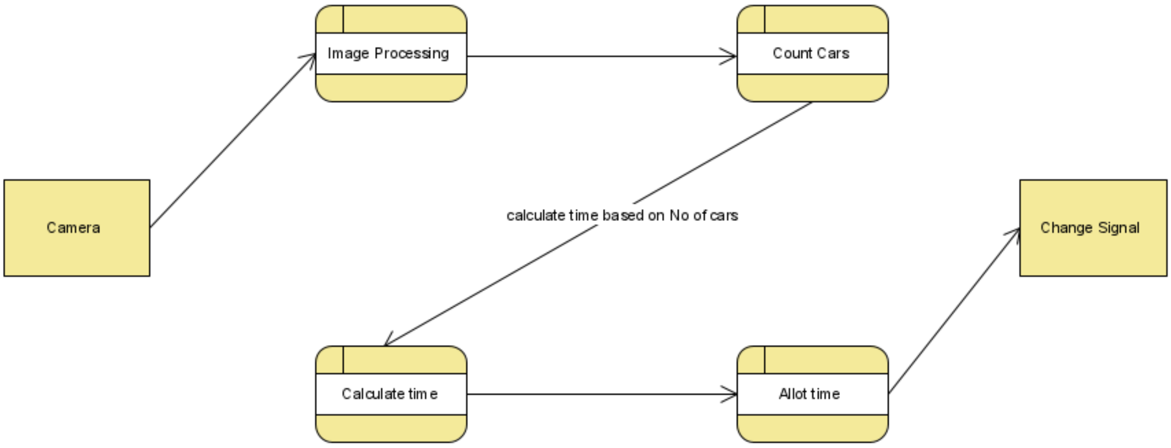


Figure 3. 3 Level 1 Dataflow diagram for DBTCS

3.3.3 Sequence Diagrams

A sequence diagram shows the interactions between objects in sequential order. It is used to describe how and in what order (sequence) the objects in a system function and interact.

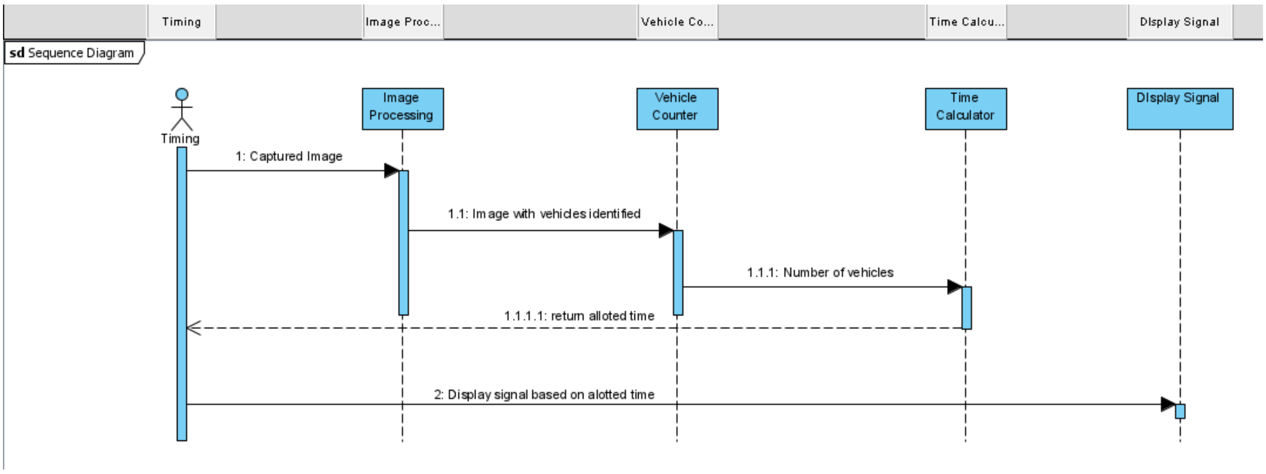


Figure 3. 4 shows the timing sequence diagram of the DBTCS

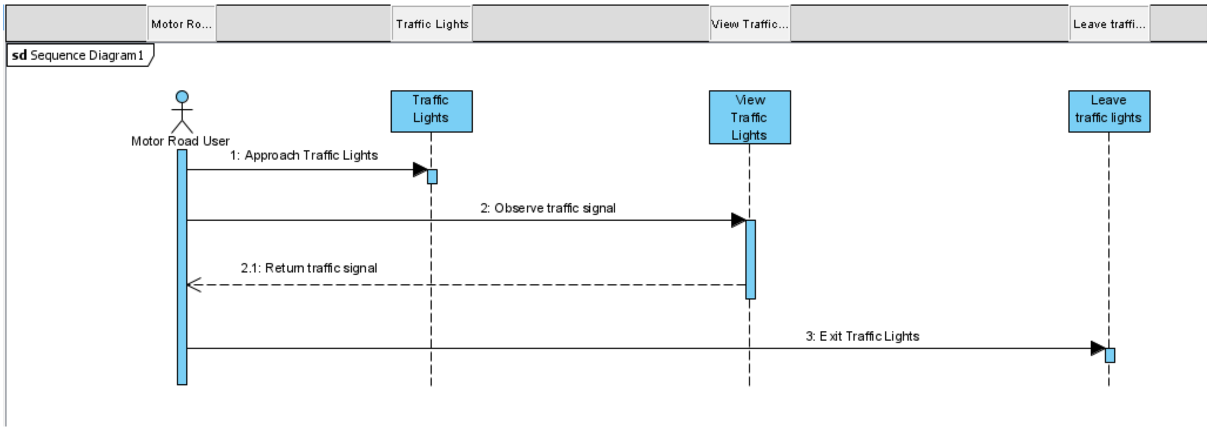


Figure 3. 5 shows the motor road user sequence diagram of the DBTCS

3.4.4 State Machine Diagram

A state machine diagram simply shows the behavior of a system by depicting its various states and transitions. It also portrays the flow of activities from a start point to an endpoint showing the different decision paths that are in existence while a state is being executed.

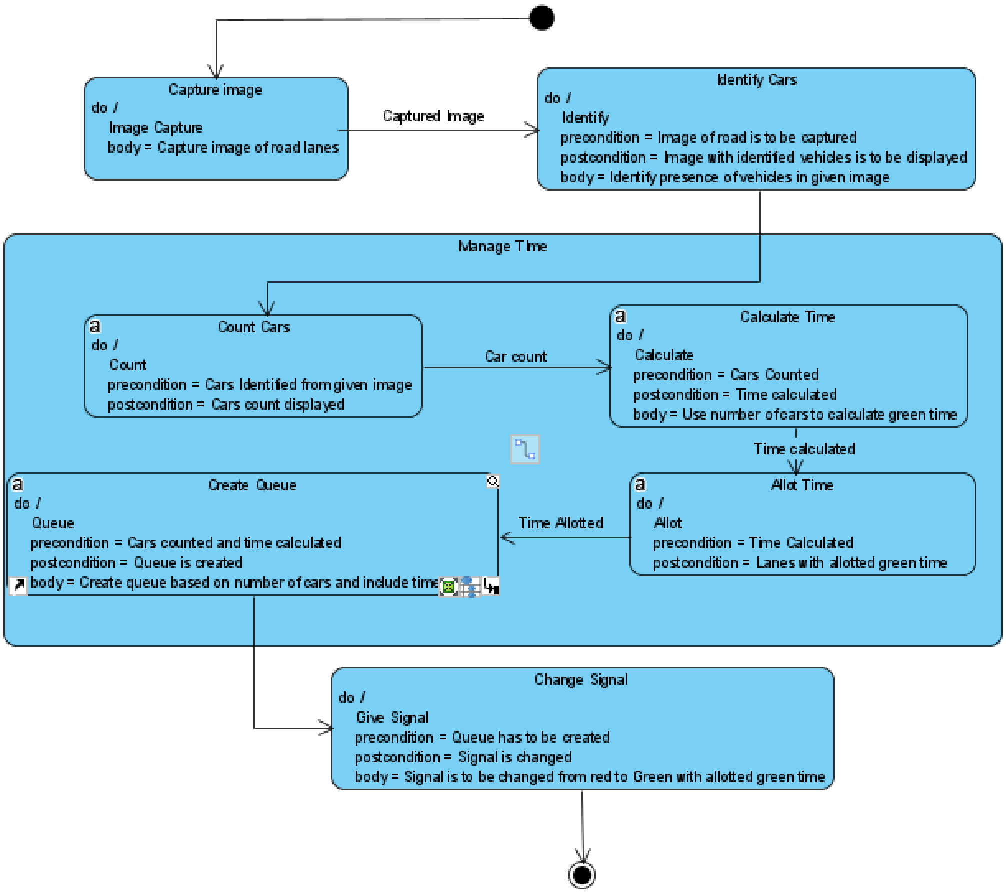


Figure 3. 6 State Machine diagram for DBTCS

3.4 Conclusion

This chapter brought forth the functional and non-functional requirements which lead to the designing of the different UML diagrams. The various diagrams help to create an effective (unified) channel between developers to understand the structure and logic of a given system.

CHAPTER 4 – DEVELOPED SYSTEM

4.1 Introduction

This chapter focuses on the various testing phases the software undergoes. The various software testing phases are processes that verify and validate that a particular software meets its specified requirements as well as help in identifying various errors or design flaws that must be fixed. The various testing phases help in the designing and development of quality software.

4.2 Unit Testing

According to Sommerville, 1977, Unit testing is simply where individual program units or objects of the system are tested. It is used to simulate all operations that are associated with the object.

4.2.1 Test Planning

Test planning is a phase where various sets of system tests are designed and developed. It demonstrates the overall coverage of the requirements. The test cases in testing planning are used to show that when an object is used as expected, its outputs are the expected results. If there are any defects, the test cases should reveal them. Below are the test cases of the system:

- If the system can perform object detection on a given image
- If the system can count detected vehicles from a given image
- If the system can calculate lane time
- If the system can generate a queue
- What happens if an image with no object is processed

Table 5.1 shows the test cases and how the system responds

Case ID	Test Case	Expected Output	Actual Outcome	Pass/Fail
1	Object Identification from image	Image with objects identified	Image with objects identified and bounding box drawn	Pass
2	Count detected vehicles in the image	vehicles from image counted and the value returned	vehicles from image counted and the value returned	Pass
3	Calculate lane Time (Based on delay value)	Lane time calculated based on the delay value	Lane time calculated Based on the delay value	Pass
4	Queue Generation (If short time is selected)	Queue generated in ascending order of number of cars in lanes	Queue generated in ascending order of number of cars in lanes	Pass
5	Queue Generation (If long time is selected)	Queue generated in descending order of number of cars in lanes	Queue generated in descending order of number of cars in lanes	Pass
6	Object detection (When no object is present in the image)	No objected is detected	No object is detected	Pass

4.3 System Testing

According to Sommerville, System testing is where all the various components are compiled together to create one full system. Its main aim is to find unanticipated errors that may be the result of component interactions and functions

4.3.1 Test Planning

Below are the test cases for the overall system:

- What happens when there are different cars in all lanes at an intersection
- What happens when there is the same number of cars in two or more lanes at an intersection
- What happens when there are no cars in all lanes at an intersection

Case ID	Test Case	Expected Output	Actual Outcome	Pass/Fail
1	Different cars in all lanes at the intersection	Green time is calculated and the queue is generated based on short or long-time variance	Green time is calculated and the queue is generated based on short or long-time variance	Pass
2	The same number of cars in two or more lanes	Green time is calculated and lanes which have similar car count are sorted by lane number thus queue is generated	Green time is calculated and lanes which have similar car count are sorted by lane number thus queue is generated	Pass
3	No cars in all lanes	Green time is calculated and the time value is 0 because no cars are present	Green time is calculated and the time value is 0 because no cars are present	Pass

4.4 User Interface Testing

User interface testing is the process of evaluating an application's user interface to ensure that there are no defects as well as to ensure that it meets its requirements specifications. It involves how the system will behave when the user selects the various configurations available. Below are the screenshots of the system:

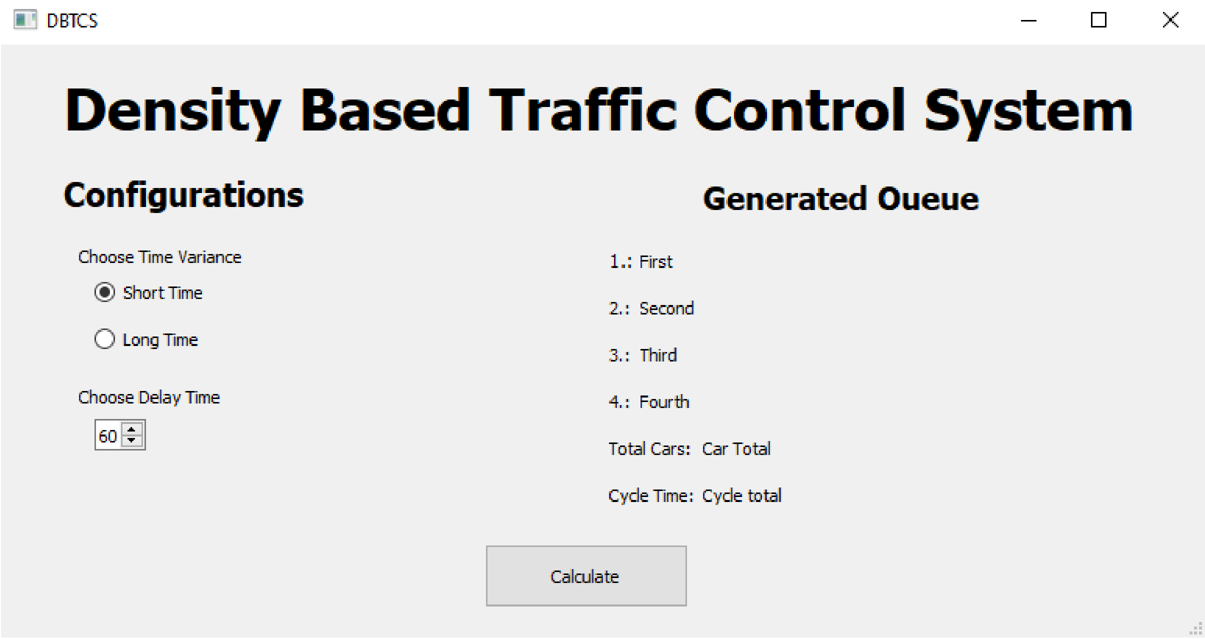


Figure 4. 1 Shows the main UI window with default selections before execution

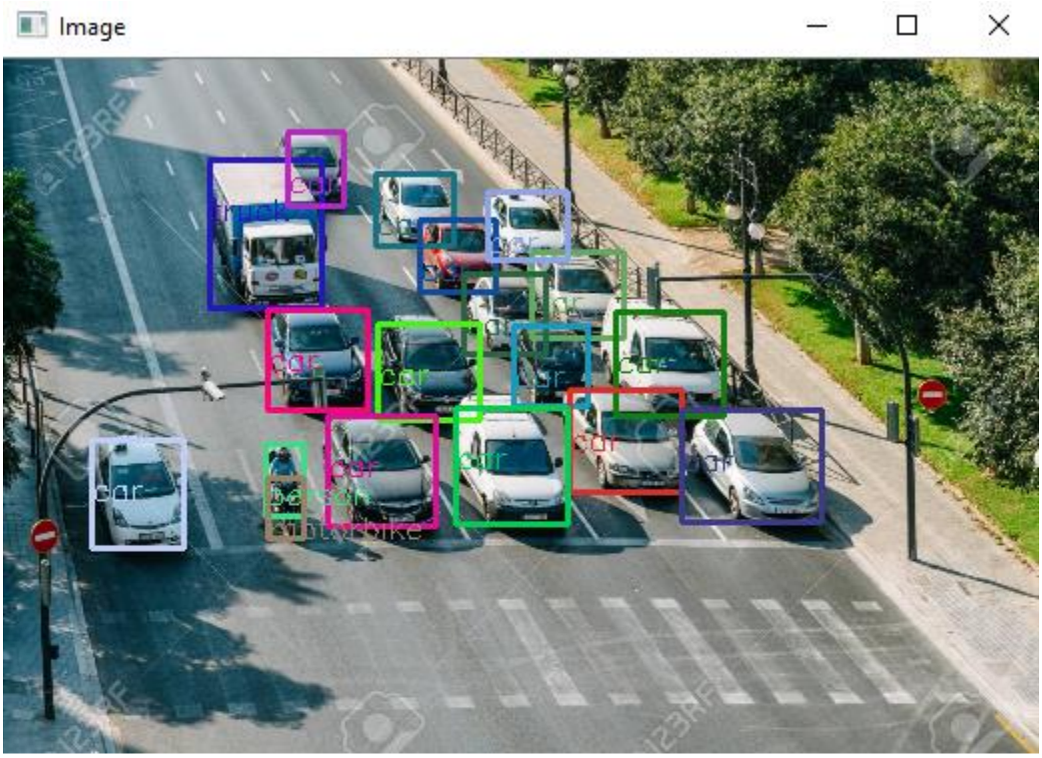


Figure 4. 2 Shows the image window which displays the processed image

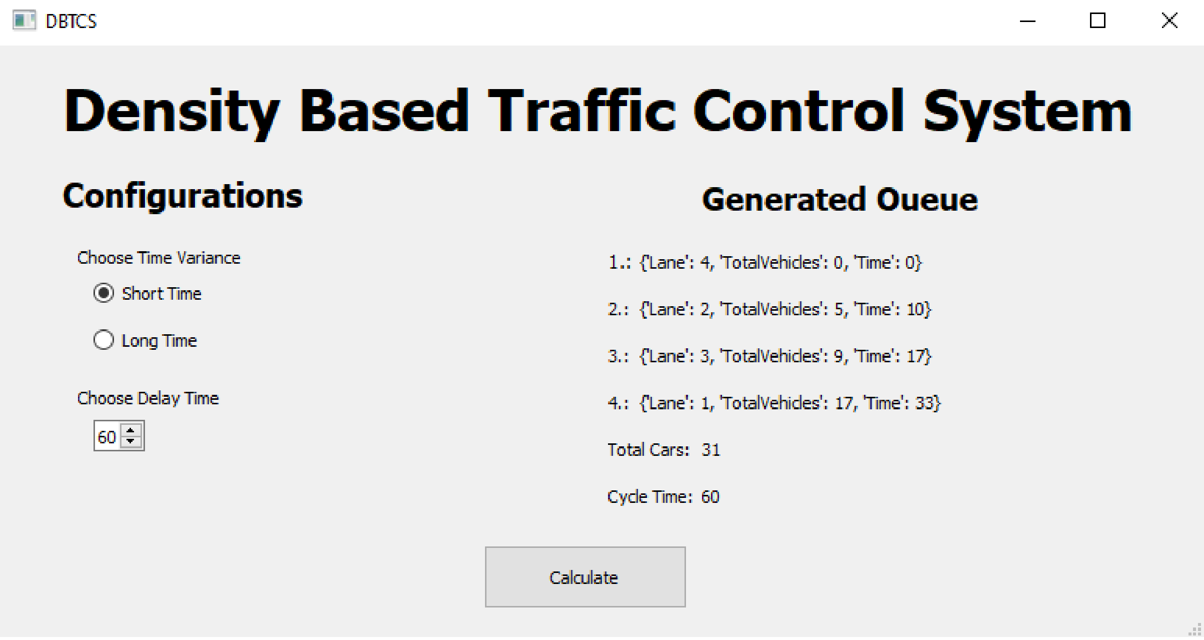


Figure 4. 3 Shows the main window after calculating the required time (with short time variance and 60 Time delay)

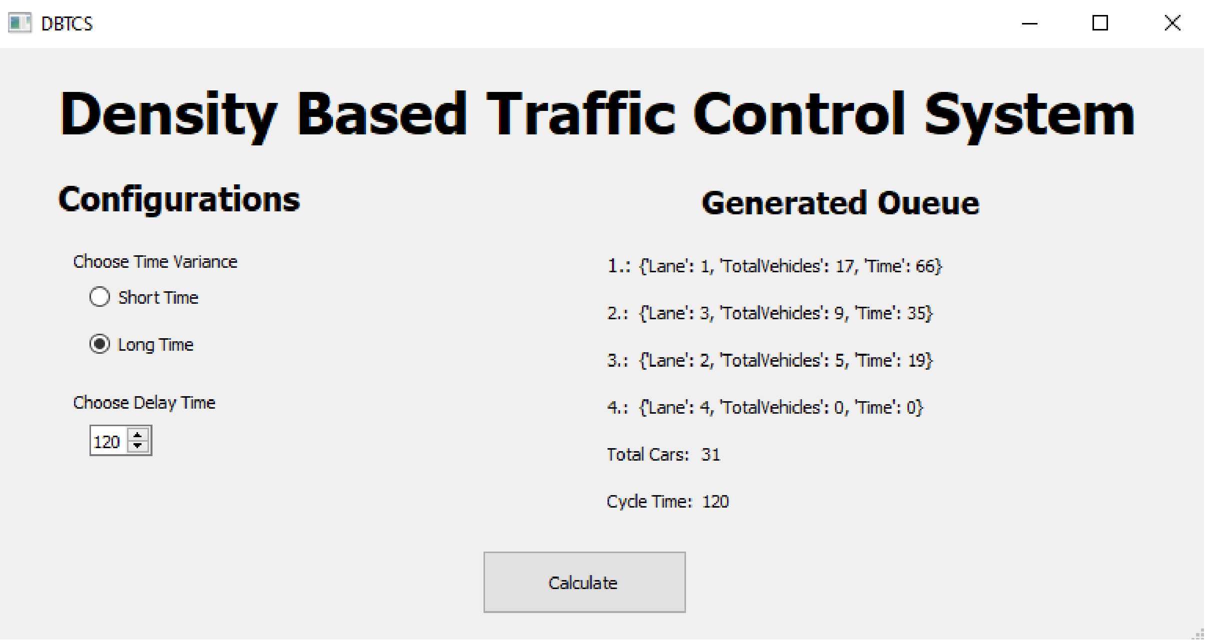


Figure 4. 4 Shows the main window after calculating the required time (with long time variance and 120 Time delay)

CHAPTER 5 – PROJECT MANAGEMENT

5.1 Introduction

Project management is simply the application of processes, methods, skills, knowledge, and experience to a project’s activities. Although good management does not ensure project success, poor management almost always leads to project failure, and such cases may lead to the software being delivered after the due date, costing more than it was originally estimated, or failing to meet the customer's expectations (Sommerville, 1977).

This chapter focuses on project risk analysis and quality management as well as providing a review of the risks that are associated with the designed system. Other sections of this chapter describe the cost model that was used for the project and the estimated costs calculated as well as outlining the schedule and work plan estimated for the system design.

5.2 Risk and Quality Management

Currently, software controls more tools and machines which are part of daily life in modern industrialized cities. Thus, the existence of risks that are directly linked to the non-functioning of software. Because of this, sufficient quality of software systems must be ensured. During software development, the project time and budget don’t allow complete coverage of the whole software functionality through software testing. Risk and the quality of the software are to be analyzed to prepare for uncertainties that might occur.

Project success is evaluated via the triple constant as shown in figure 4. 1 below:

Figure 5. 1 Triple Constant Model (Van Wyngaard et al., 2012)

Without the effective management of the triple constraint as an interrelated system, projects run the risk of becoming separated from the purpose (Van Wyngaard et al., 2012).

5.2.1 Risk Management

A risk is defined as a bundle of future uncertain events with a probability of occurrence and potential for loss. Risk management is defined as a planned and structured process aimed at helping the project team make the right decision at the right time to identify, classify, quantify the risks and then to manage and control them (Srinivas, 2018). It should begin at the outset of program development and continue throughout the program's entire life cycle project to help ensure project success or account for uncertainties.

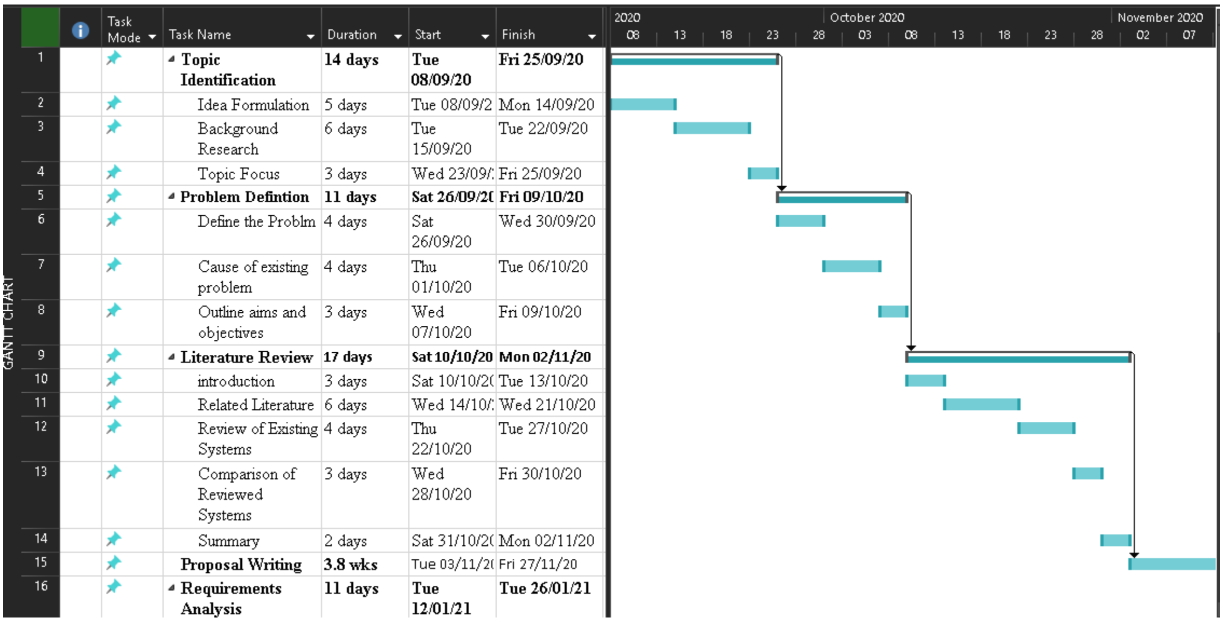
5.3 Effort Costing Model

The Constructive Cost Model, also referred to as COCOMO, is a well-documented and widely accepted algorithmic model for effort estimation. COCOMO was introduced by Boehm in 1981. This model consists of mathematical equations that identify the developed time, the effort, and the maintenance effort (F. & Aljahdali, 2013). The constructive Cost Model has three variants. These are basic, intermediate, and detailed. The main parameter used for computations is the size of the project. The size is represented in terms of lines of code (LOC) or a thousand lines of code (KLOC).

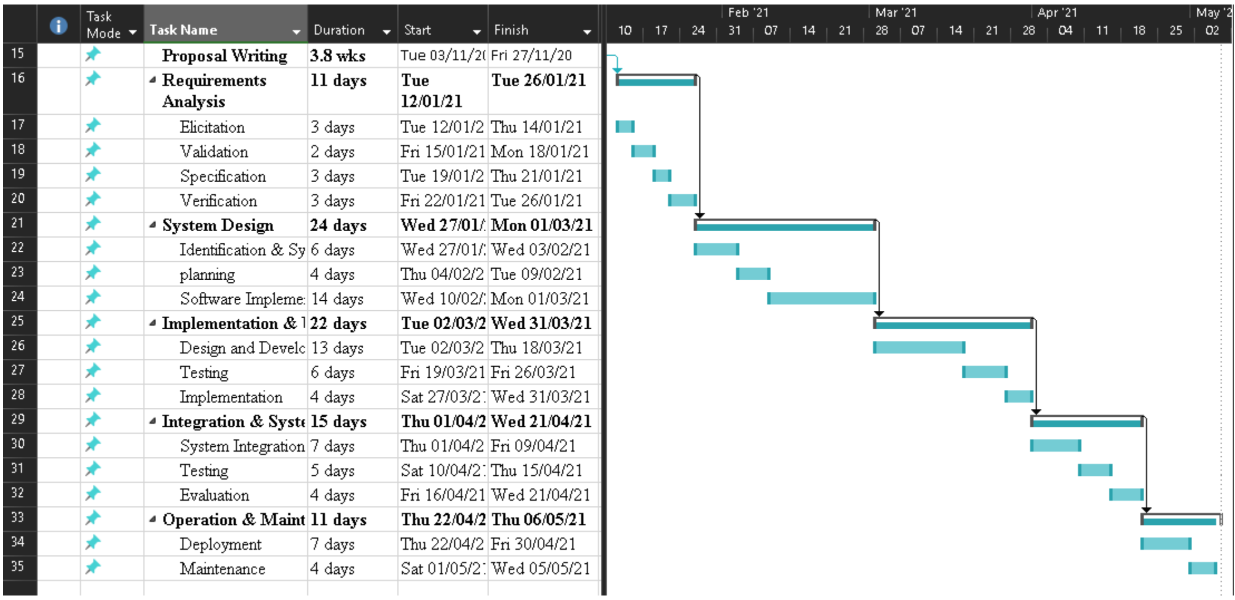
$$E = a(KLOC)^b$$

Where E represents the software effort computed in person-month, KLOC denotes the size of the code (kilo-lines of code), and A, B represents the COCOMO model parameters. The value of A and B depends on the model of a software project. COCOMO model has three types, depending upon project size. These models are the Organic, Semi-detached, and Embedded model (Sachan et al., 2016).

5.4 Scheduling and Work plan



	Task Mode	Task Name	Duration	Start	Finish
1		**Topic Identification**	**14 days**	**Tue 08/09/20**	**Fri 25/09/20**
2		Idea Formulation	5 days	Tue 08/09/2	Mon 14/09/20
3		Background Research	6 days	Tue 15/09/20	Tue 22/09/20
4		Topic Focus	3 days	Wed 23/09/:	Fri 25/09/20
5		**Problem Defintion**	**11 days**	**Sat 26/09/2(**	**Fri 09/10/20**
6		Define the Problm	4 days	Sat 26/09/20	Wed 30/09/20
7		Cause of existing problem	4 days	Thu 01/10/20	Tue 06/10/20
8		Outline aims and objectives	3 days	Wed 07/10/20	Fri 09/10/20
9		**Literature Review**	**17 days**	**Sat 10/10/20**	**Mon 02/11/20**
10		introduction	3 days	Sat 10/10/2(	Tue 13/10/20
11		Related Literature	6 days	Wed 14/10/:	Wed 21/10/20
12		Review of Existing Systems	4 days	Thu 22/10/20	Tue 27/10/20
13		Comparison of Reviewed Systems	3 days	Wed 28/10/20	Fri 30/10/20
14		Summary	2 days	Sat 31/10/2(	Mon 02/11/20
15		**Proposal Writing**	**3.8 wks**	Tue 03/11/2(	Fri 27/11/20
16		**Requirements Analysis**	**11 days**	**Tue 12/01/21**	**Tue 26/01/21**



	Task Mode	Task Name	Duration	Start	Finish
15		**Proposal Writing**	**3.8 wks**	Tue 03/11/2(	Fri 27/11/20
16		**Requirements Analysis**	**11 days**	**Tue 12/01/21**	**Tue 26/01/21**
17		Elicitation	3 days	Tue 12/01/2	Thu 14/01/21
18		Validation	2 days	Fri 15/01/21	Mon 18/01/21
19		Specification	3 days	Tue 19/01/2	Thu 21/01/21
20		Verification	3 days	Fri 22/01/21	Tue 26/01/21
21		**System Design**	**24 days**	**Wed 27/01/**	**Mon 01/03/21**
22		Identification & Sy	6 days	Wed 27/01/:	Wed 03/02/21
23		planning	4 days	Thu 04/02/2	Tue 09/02/21
24		Software Impleme:	14 days	Wed 10/02/:	Mon 01/03/21
25		**Implementation & I**	**22 days**	**Tue 02/03/2**	**Wed 31/03/21**
26		Design and Develc	13 days	Tue 02/03/2	Thu 18/03/21
27		Testing	6 days	Fri 19/03/21	Fri 26/03/21
28		Implementation	4 days	Sat 27/03/2:	Wed 31/03/21
29		**Integration & Syste**	**15 days**	**Thu 01/04/2**	**Wed 21/04/21**
30		System Integration	7 days	Thu 01/04/2	Fri 09/04/21
31		Testing	5 days	Sat 10/04/2:	Thu 15/04/21
32		Evaluation	4 days	Fri 16/04/21	Wed 21/04/21
33		**Operation & Maint**	**11 days**	**Thu 22/04/2**	**Thu 06/05/21**
34		Deployment	7 days	Thu 22/04/2	Fri 30/04/21
35		Maintenance	4 days	Sat 01/05/2:	Wed 05/05/21

Figure 5. 2 Schedule and work plan for the designed system

5.5 Summary

This chapter is focused on effort cost as well as risk and quality management whose function is to assist in the assessment of risks that might be encountered and means of avoiding them and effort costs which help to determine the time and effort that would be required to complete the project according to the schedule. COCOMO help to give an estimate of the software costs while the Gantt chart helps outline the documentation and schedule for the development methodologies outlined.

CHAPTER 6 – CRITICAL EVALUATION

6.1 Introduction

The main purpose of this chapter is to outline the critical evaluation of the project. It will bring to attention the purpose of the research, challenges encountered during the development process, lessons learned, and lastly future recommendations of the system.

6.2 Reason for Undertaking the Project

The main reason the project was undertaken was to help reduce traffic congestion as well as the wait times at traffic lights to quicken or enhance the smooth flow of traffic.

6.3 Main Learning Outcome

The following are the lessons learned from the undertaking of this project:

- Value of Research

  The design and development of this project highlighted the importance and value of research. It outlined that for a project to be successful, research has to be carried to know how and where various components of a system can be put together.

- Hard work, Patience and Perseverance

  The design and development of this project also highlighted the importance of hard work, perseverance, and patience. For a system or project to be built, patience, hard work, and perseverance are to be put in perspective so that goals, deadlines, and targets can be achieved as doing things in a rush by cutting corners tends to yield errors and mistakes which can or could have been avoided.

- Planning and Time Management

  The undertaking of this project taught me the importance of planning and time management. For a project to be built, a plan or schedule has to be designed outlining the various tasks and deadlines to be met.

- Problem-Solving

  The development of this project taught me various on how to face and address various barriers and efficient ways of achieving problem-solving.

6.4 Challenges Encountered

Throughout the development of the system, the python programming language and its various libraries had to be learned to design the system. Image processing in a complex field that helps machines understand the human-like vision and as such its various complex techniques had to be understood and to also design the system.

Another challenge that was encountered was the finding of the collection of test data to use to evaluate the system. Due to the Covid-19 pandemic, movements were limited and as such, it was a challenge to collect test data and various intersections.

6.5 Future Works

The developed version of the density-based traffic control system will be able to achieve the fundamental functionalities of traffic lights. However, there is room for upgrades and other systems can be integrated. The following are future works that can be implemented:

- A red-light violation feature can be added to capture vehicles that speed past an intersection when the lights are red.
- The system can be networked so that it can be controlled or modified from a remote location.
- A feature can be added to encompass pedestrians crossing
- Implementation of the system using night vision cameras/thermal cameras
- Database technologies can be added to log the data for statistical purposes
- Incase of bad weather a module can be implemented to switch from the density based to the fixed time allocation.

6.6 Conclusion

In conclusion, it can be seen that for a project to be conducted there are various steps and procedures. These various steps and procedures can produce challenges as well be learning curves. This chapter outlined the various challenges faced, learning outcomes, and new skills or attributes gained by the developer.

CHAPTER 7 – CONCLUSION

In conclusion, the Density Based Traffic Control System is developed to help replace the existing system which is the Traffic Lights with a standard allotted time. The system to helps reduce traffic congestions as well as improve traffic flow. The implementation of image processing technology helps in the identification of traffic density and decision-making of time to be allotted for a particular lane.

This project is aimed at getting rid of the problems that were identified in the initial analysis of the thesis. Some of the problems identified were traffic congestion, unnecessary wait times at traffic intersections when there is no traffic and eliminating the standard allotted time for traffic lights. These problems are faced by most if not all road users.

Therefore, the system is designed to eliminate these problems. Some solutions that will be applied to eliminate these problems include the use of image processing technology as well as eliminating the standard allotted time. The indicated solutions help improve traffic flow as well as get rid of traffic congestion which is the main aim of the project.

REFERENCES

Bhaskar, L., Sahai, A., Sinha, D., Varshney, G., & Jain, T. (2016). Intelligent traffic light controller using inductive loops for vehicle detection. *Proceedings on 2015 1st International Conference on Next Generation Computing Technologies, NGCT 2015*, *September*, 518–522. https://doi.org/10.1109/NGCT.2015.7375173

Dabbagh, M., Lee, S. P., & Parizi, R. M. (2016). Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches. *Soft Computing*, *20*(11), 4497–4520. https://doi.org/10.1007/s00500-015-1760-z

Dennis, A., Barbara, Haley, W., & Roberta M, R. (2015). Systems Analysis & Design. In *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis* (Vol. 53, Issue 9). http://publications.lib.chalmers.se/records/fulltext/245180/245180.pdf%0Ahttps://hdl.handle.net/20.500.12380/245180%0Ahttp://dx.doi.org/10.1016/j.jsames.2011.03.003%0Ahttps://doi.org/10.1016/j.gr.2017.08.001%0Ahttp://dx.doi.org/10.1016/j.precamres.2014.12

Gunda, S. G. (2008). *Requirements Engineering : Elicitation Techniques*. 40.

Harris, A., Lang, M., Oates, B., & Siau, K. (2006). Systems analysis & design: An essential part of IS education. *Journal of Information Systems Education*, *17*(3), 241.

How to Perform Object Detection With YOLOv3 in Keras. (n.d.). Retrieved May 26, 2021, from https://machinelearningmastery.com/how-to-perform-object-detection-with-yolov3-in-keras/

Jadhav, A. D., Madhuri, B., & Ketan, T. (2015). *Intelligent Traffic Light Control System Based Image Intensity*. *September*, 84–87.

Kerala Police, 2019. *Traffic Officer Manually Controlling Traffic*. [image] Available at: <https://old.keralapolice.gov.in/wings/specialized-wings/traffic-police> [Accessed 1 November 2020].

Malambo, C. & Chomba, G., 2018. *Traffic Officer Controlling Traffic In Bad Weather*. [image] Available at: <https://zambiareports.com/2018/05/17/police-officer-rainstraffic-control/cop-rains/> [Accessed 1 November 2020].

Michels, B. (2014). Traffic Light History Abstracts. *Journal of Chemical Information and Modeling*, *53*(9), 1689–1699.

Naveenkumar, M. (2016). *OpenCV for Computer Vision Applications*. *March 2015*, 52–56.

Nosrati, M. (2011). Python: An appropriate language for real-world programming. *World Applied Programming*, *12*, 110–117.

Panjwani, M., Tyagi, N., Venkata Lakshmi Narayana, K., & Shalini, D. (2014). Smart traffic control using image processing. *International Journal of Applied Engineering Research*, *9*(16), 3429–3442.

Parr, S. A., Herrera, N., Wolshon, B., & Smith, T. (2020). Effect of manual traffic control on evacuation time estimates. *Transportation Research Record*, *2674*(9), 809–819. https://doi.org/10.1177/0361198120932165

Parr, S., & Wolshon, B. (2015). *Manual traffic control for planned special events and emergencies 7*. *7*(2).

Pashupatimath, S. G. (2018). Different Techniques Used in Traffic Control System : An Introduction. *International Journal of Engineering Research & Technology (IJERT), 6*(03), 1–4.

Sachan, R. K., Nigam, A., Singh, A., Singh, S., Choudhary, M., Tiwari, A., & Kushwaha, D. S. (2016). Optimizing Basic COCOMO Model Using Simplified Genetic Algorithm. *Procedia Computer Science*, *89*, 492–498. https://doi.org/10.1016/j.procs.2016.06.107

Scherfgen, D., & Sc, M. (2015). *Introduction to OpenCV (Part 1)*. *Part 1*.

Shinde, Y., & Powar, H. (2017). *Intelligent Traffic Light Controller Using IR Sensors for Vehicle Detection*. *Ncetete*, 2393–2395. https://doi.org/10.17148/IARJSET/NCETETE.2017.28

Sommerville, I. (1977). Software engineering. In *Science* (Vol. 195, Issue 4283). https://doi.org/10.1126/science.195.4283.1199Dabbagh, M., Lee, S. P., & Parizi, R. M. (2016). Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches. *Soft Computing*, *20*(11), 4497–4520. https://doi.org/10.1007/s00500-015-1760-z

Sommerville, I. (2006). Software Engineering (8th Ed.). In *Software Engineering (8th Ed.)*.

System Analysis and Design - Overview - Tutorialspoint. (n.d.). Retrieved May 15, 2021, from https://www.tutorialspoint.com/system_analysis_and_design/system_analysis_and_design_overview.htm

Srinivas, K. (2018). Process of Risk Management K. Srinivas. *IntechOpen*, 1–16. https://www.intechopen.com/books/advanced-biometric-technologies/livenessdetection-in-biometrics

Udoakah, Y. N., & Okure, I. G. (2018). Design and implementation of a density-based traffic light control with surveillance system. *Nigerian Journal of Technology*, *36*(4), 1239. https://doi.org/10.4314/njt.v36i4.34

Van Wyngaard, C. J., Pretorius, J. H. C., & Pretorius, L. (2012). Theory of the triple constraint -A conceptual review. *IEEE International Conference on Industrial Engineering and Engineering Management*, *December*, 1991–1997. https://doi.org/10.1109/IEEM.2012.6838095

Varambally, Samhita & Shahi, Prachi & Rajesh, A. (2018). *Intelligent Traffic Density Monitoring Using RFID*. *March*.

Yousef, A., Shatnawi, A., & Latayfeh, M. (2019). Intelligent traffic light scheduling technique using calendar-based history information. *Future Generation Computer Systems*, *91*(August), 124–135. https://doi.org/10.1016/j.future.2018.08.037

Yousef, K. M., Al-Karaki, J. N., & Shatnawi, A. M. (2010). Intelligent traffic light flow control system using wireless sensors networks. *Journal of Information Science and Engineering*, *26*(3), 753–768. https://doi.org/10.6688/JISE.2010.26.3.2

Zima, D. (2015). Modern Methods of Software Development. *Task Quarterly*, *19*(4), 481–493.