FUZZY LOGIC AGGREGATION OF WIRELESS SENSOR NETWORK DATA FOR SMART TRAFFIC LIGHT CONTROL

PRINCESS ROXANNE HAWI OKORE

MASTER OF SCIENCE

(Computer Systems)

JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY

2015

Fuzzy Logic Aggregation of Wireless Sensor Network Data for Smart Traffic Light Control

Princess Roxanne Hawi Okore

A thesis submitted in partial fulfilment for the degree of the Masters of Science in Computer Systems in Jomo Kenyatta University of Agriculture and Technology.

2015

DECLARATION

This thesis is my original work and has not been presented for a degree in any other University

Signature ……Date……

Roxanne Hawi Okore

This thesis has been submitted for examination with our approval as University Supervisors:

Signature ……Date……

Dr. George Okeyo
JKUAT, Kenya

Signature ……Date……

Dr. Michael Kimwele
JKUAT, Kenya

DEDICATION

This work is dedicated to my Mother and Father.

ACKNOWLEDGEMENT

This project would not have been possible without the support of many people. Thanks to my supervisors, George Okeyo and Michael Kimwele, who read my numerous revisions; and helped me to successfully complete this thesis. Also thanks to the Jomo Kenyatta University of Agriculture and Technology for providing me with educational materials needed for the completion of this work. Finally special thanks to my mother who offered a lot of financial guidance; and endured this long process with me, always offering support and love.

TABLE OF CONTENTS

DEDICATION ..... **ii**

ACKNOWLEDGEMENT ..... **iii**

LIST OF FIGURES ..... **vi**

LIST OF TABLES ..... **vii**

LIST OF ABBREVIATIONS/ACRONYMS ..... **viii**

ABSTRACT ..... **x**

CHAPTER ONE **1**

INTRODUCTION **1**

1.1 Background 1

1.2 Problem Statement 3

1.3 Objectives 4

1.4 Research Questions 5

1.5 Justification 5

1.6 Knowledge Contributions 5

1.7 Scope 6

1.8 Outline of the Thesis 7

CHAPTER TWO **8**

LITERATURE REVIEW **8**

2.1 Sensing systems 8

2.1.1 Sensor Networks 8

2.1.2 Wireless Sensor Networks (WSN) 10

2.2 Existing Traffic Control Systems 11

2.2.1 Static systems 11

2.2.2 Sensor based smart systems 12

2.3 Research gap 22

2.4 Conclusion 22

CHAPTER THREE **24**

METHODOLOGY **24**

3.1 Research Methodology 25

3.1.1 Research Design 25

3.1.2 Research Methods/Process 25

3.1.3 Data Collection Tools 26
3.2 Development Tools 26
3.2.1 Process Model 26
3.2.2 Development Procedure 27
3.3 Conclusion 50
CHAPTER FOUR....52
RESEARCH FINDINGS AND DISCUSSION 52
4.1 Experimental Setup 52
4.1.1 Metrics for Evaluation/Performance measures....53
4.2 Observations 54
4.3 Simulation Results....62
4.4 Discussion....64
4.5 Conclusion 65
CHAPTER FIVE 67
CONCLUSION AND RECOMMENDATIONS FOR FUTURE WORK 67
5.1 Conclusion 67
5.2 Recommendation for future work 68
REFERENCES 70
APPENDIX 75

LIST OF FIGURES

Figure 2.1: High level description of traffic time manipulation algorithm (TSTMA)[Yousef *et al.*, 2010] 20
Figure 3.1: Research Process Flowchart 25
Figure 3.2: Conceptual Framework for Smart Traffic Light Control Using Fuzzy Logic and WSN 29
Figure 3.3: Class Diagram for the Developed Software System 30
Figure 3.4: Interface for our Persistent Storage for data retrieval/reference 35
Figure 3.5: Smart Traffic Light Routing Algorithm 36
Figure 3.6: Screenshot of the system's interface in Java 37
Figure 3.7: Graphical representation of TriMF 38
Figure 3.8: Graphical representation of input variable WT_T 40
Figure 3.9: Graphical representation of input variable TQ_T 40
Figure 3.11: Graphical representation of our qtFuzzylite application 41
Figure 3.10: Graphical representation of output variable PD 41
Figure 3.13: FLC Process 42
Figure 3.14: Graphical representation of fuzzification 42
Figure 3.15: FLC rules for Traffic Light Controller 45
Figure 3.16: Activating the antecedent 46
Figure 3.17: Graphical representation of modification of consequents 47
Figure 3.18: Graphical representation of accumulation of consequents 47
Figure 3.19: Graphical representation of centroid defuzzifier 48
Figure 3.20: Screenshot of our model in operation 50
Figure 4.1: Average distribution of WT and PD with relation to TQ across the 5 rounds 61
Figure 4.2: Average PD distribution for all lanes in each round 62

LIST OF TABLES

Table 3.1: Matrix representation of the fuzzy set rules 46
Table 4.1: R1 data collected from the four approaches for TQ and WT 55
Table 4.2: R2 data collected from the four approaches for TQ and WT 55
Table 4.3: R3 data collected from the four approaches for TQ and WT 56
Table 4.4: R4 data collected from the four approaches for TQ and WT 56
Table 4.5: R5 data collected from the four approaches for TQ and WT 56
Table 4.6: R1 computed PD values from the four approaches 57
Table 4.7: R2 computed PD values from the four approaches 57
Table 4.8: R3 computed PD values from the four approaches 58
Table 4.9: R4 computed PD values from the four approaches 58
Table 4.10: R5 computed PD values from the four approaches 59
Table 4.11: Average results of the data collected on TQ, WT and PD for all four lanes in 76 cycles for the five rounds 60
Table 4.12: Description summary of the errors encountered 63
Table 4.13: Comparison of highest and lowest TQ and WT averages with relation to highest and lowest PD averages for all 5 rounds.....63

LIST OF ABBREVIATIONS/ACRONYMS

AI	–	Artificial Intelligence
ANN	–	Artificial Neural Networks
ATMS	–	Advanced Traffic Management System
BS	–	Base Station
CBT	–	Campaign for Better Transport
CCTV	–	Closed Circuit Television
FES	–	Fuzzy Expert System
FLC	–	Fuzzy Logic Controller
H	–	High
ITS	–	Intelligent Transportation System
IR	–	Infrared
L	–	Low
M	–	Medium
NN	–	Neural Networks
PD	–	Priority Degree
R1	–	Round One
R2	–	Round Two
R3	–	Round Three
R4	–	Round Four
R5	–	Round Five
RCU	–	Road Control Units
RFID	–	Radio Frequency Identification
SMS	–	Short Message Service
STCS	–	Smart Traffic Control System
TQ	–	Traffic Quantity
VA	–	Vehicle Actuated
VLSI	–	Very Large Scale Integration
WT	–	Waiting Time

ABSTRACT

The steady increase in the number of vehicles on the road has increased traffic congestion in most urban cities. Inadequate space and funds for the construction of new roads has prompted scholars to investigate other solutions to traffic congestion. Most popular approach in use is the traffic light system. Adopted strategies include; static and vehicle actuated (VA) lights. However, these models have limitations. The static model does not take into account the non-uniform and dynamic nature of traffic because they do not operate with real time data. VA lights were an attempt to improve the static model; they activate a change in light signal when cars are present. However, they only detect presence of cars; they do not count number of cars. To help solve these weaknesses, this research presented a novel approach to traffic routing. Our approach uses smart traffic control systems (STCS) to make traffic routing decisions. STCS use real time data and mimic human reasoning thus prove promising in vehicle traffic control. We present a smart traffic light controller using fuzzy logic and wireless sensor network (WSN). The approach is designed for an isolated four way roundabout. It employed fuzzy logic to control the lights and determine how the green light will be assigned for each approach. The WSN collected the traffic data in real time. This data is aggregated and fed into a fuzzy logic controller (FLC) in form of two inputs – traffic quantity (TQ) and waiting time (WT) for each approach. Based on the inputs, the FLC then computes an output priority degree (PD) that controls green light assignment. Using the PD, an algorithm is formulated that assigns green light to the lane with highest PD. The cycle continues until all approaches get green. Given the practical nature of the thesis, applied research was the core methodology used. This research design made it possible to gather necessary data, analyze it and develop a solution to address the weaknesses of the current traffic light controllers. To test and analyze the approach, we designed and simulated a model of a traffic light in Java. This provided a virtual representation of the proposed approach. The test bed provided 94.7% accuracy. The results further demonstrated that; a smart traffic light controller has better performance than a static one; when it comes to reducing average WT at intersections. Our approach recorded an average WT of 2.985 minutes while the fixed had an average WT of 8.955 minutes. It also showed that STCS can fairly manage traffic at four-way intersections. This is proven by the fact that our average WT was 2.985 minutes against the 6 minute maximum WT limit; and the average PD was 5.091, against a PD range of 2.500 and 7.500. Lastly the results indicate that a smart traffic

light controller can effectively replace an experienced traffic officer managing traffic at a roundabout.

CHAPTER ONE

INTRODUCTION

1.1 Background

Road transportation remains a key socioeconomic infrastructure today even with the increased advancements in air, rail and water transport. The steady increase in the number of vehicles on the road has increased traffic congestion in most urban cities around the world. Owing to this traffic congestion, a number of challenges arise. To a commuter or traveller, congestion means lost time, missed opportunities, and frustration. To an employer, congestion means lost worker productivity, trade opportunities, delivery delays, and increased costs (Wen, 2008).

One approach most countries are taking to address this issue is the expansion of roadways. This approach still comes with its share of challenges. Demolition of older roads can be quite costly. Most urban cities lack the free space required for such a venture. Even with the improvements in road infrastructure, it is evident that the rate at which travellers buy vehicles has surpassed that of new infrastructure development. Also due to expansions, roads are able to serve more vehicles; consequently utilising the additional capacity.

In 1942, Robert Moses, observed that the highways he built around New York in 1939 were in the long run generating greater traffic congestion than had existed previously. This observation prompted scholars to further investigate on the causal relationship between traffic congestion and road supply. A recent study done at University of Toronto by Duranton and Turner (2011), confirms the 'fundamental law of highway congestion' suggested by Downs (1962) that an extension of roads is met with a proportional increase in traffic.

Locally, Kenyan commuters also face traffic jams. In fact according to the IBM (2011), Kenya was ranked 4th among the 20 cities in the world with the worst traffic

jams, with 35% of drivers in Nairobi reporting that they have spent three hours or more in traffic. Kenya's Vision 2030 launched in 2006 has prompted the expansion of highways and bypasses country wide (GoK, 2007). However, this approach has not proven to be so effective in managing traffic in Nairobi. This is consistent with Downs' law (1962) that increasing road supply invariably increases vehicle traffic. From this it is evident that Kenya also needs new traffic management alternatives besides the traditional approach of just building more roads. Christian Scholsser Chief of Urban Transport with UN-Habitat, suggests that smarter transport transcends infrastructure. He proposes that physical infrastructure needs to be combined with new ways of thinking and new technologies (IBM, 2012).

There has been little use of smart technology in managing traffic flow in developing countries. Most popular approach in use in Kenya is the static traffic light system. The main reason being, they are easy to implement and inexpensive. The static model's traffic lights timing and switching patterns are predetermined despite prevailing traffic conditions. They behave in the same way during the peak and normal hours and even during special situations such as parade, accident, floods (Runyoro *et al*., 2014). It does not take into account the non-uniform and dynamic nature of traffic because they do not operate with real time data. Consequently, this lack of adaptive strategies in these devices does very little in improving the road network performance and the traffic congestion levels (Kumar, 2011). This was demonstrated in Kenya when the country recently experienced a setback when the Nairobi County Government pulled out policemen from the roads to test the newly automated static traffic lights. The lights had an additional counter feature – that counts down from one light to the other (Ndonga, 2014). This meant that the motorists could then know how long to wait before moving. However, this operation was not successful; there was a traffic gridlock in most parts of the city with some motorists spending close to 4 hours in traffic. (Hawi, *et al*., 2015).

This thesis strives to introduce a novel approach to traffic management – fuzzy logic aggregation of wireless sensor network data for smart traffic light control. Our approach is designed to alleviate the weaknesses of the current traffic control

systems. This is achieved by integrating advanced traffic management system (ATMS). This is one of the four subsystems of intelligent transportation system (ITS). ATMS performs tasks such as surveillance, control and management of freeway and arterial networks, intersection traffic light control and congestion and incident management (Patel *et al.*, 2001). Our research interest being developing countries particularly Kenya; incorporating ITS as a whole would be too large and complex hence very costly (Yokota, 2004). Therefore our approach mainly focused on integrating ATMS in traffic lights so as to dynamically route traffic at a four-way isolated junction. We employ fuzzy logic to control the traffic lights and determine how the green light will be assigned for each approach. The wireless sensor network (WSN) collects traffic data in real time. This data is aggregated and then fed into a fuzzy logic controller (FLC) engine in form of two inputs – traffic quantity (TQ) and waiting time (WT) for each lane. Based on the inputs, the FLC then computes an output priority degree (PD) that controls green light assignment in traffic lights. Using the PD, an algorithm is formulated that assigns green light to the lane with highest PD. Once a lane gets green light, it has to wait till all other lanes get green light before being assigned green again. This is referred to as a cycle. To test and analyze our approach, we design a simulation in Java to virtually represent the functions of the WSN and FLC. The results from our simulations demonstrate that our approach if implemented can effectively reduce traffic congestion and replace traffic officers managing traffic at a roundabout.

1.2 Problem Statement

The gravity of road traffic congestion has worsened worldwide particularly in developing countries. A number of factors contribute to this traffic problem. Firstly, there is a steady increase in number of vehicles on the roads. The Kenya Revenue Authority registered 205,841 vehicles in 2011 up from 161,813 vehicles in 2009 (KIPPRA, 2013). Emerging economies are resorting to expand and build new roads as a solution. However, this approach is not sustainable due to costs. Also new roads generate new journeys and increase traffic pressure on the surrounding local network (CBT, 2012). Consequently, the road infrastructure is still not able to keep up with the growing traffic demand.

Another factor is use of old traffic control technologies. The major traffic control systems used in developing countries are pre-timed. Unfortunately, the nature of traffic is uncertain and non-uniform. Consequently use of these predetermined controllers becomes inefficient when traffic demand and fluctuation is high. (Kumar, 2011). For instance, the current traffic lights in Nairobi use timers and counters; in which data is pre-programmed into the light's chip/PLC. It does not matter whether at a particular period of time route one normally has more cars than route two; the green light allocation time will still remain the same. This will definitely cause some congestion, especially on route one.

Lastly, poorly planned road networks are also a major contributor to the increased road traffic congestion levels in Kenya. Traffic junctions are the major causes of delays in most road networks because they are where most roads converge and vehicles need to traverse different directions. Lack of existing sensor networks synchronising the traffic lights at different junctions leads to a mismatch in flow of traffic of vehicles from one intersection to another.

1.3 Objectives

The main goal is to develop an approach to smart traffic light control to autonomously manage traffic at a four way roundabout.

The following are the specific objectives that will help achieve this goal:

i) To investigate the use of sensor networks in vehicle traffic management.
ii) To identify techniques for aggregating traffic data for traffic light control.
iii) To formulate a fuzzy logic based architecture that aggregates traffic sensor data fed from a WSN.
iv) To develop an algorithm that uses data from the WSN and FLC to dynamically allocate traffic light signal to each lane.
v) To evaluate the effectiveness of using fuzzy logic and WSN in traffic control systems.

1.4 Research Questions

i) How does the current vehicle traffic management system operate at roundabouts?
ii) What techniques will be employed to gather traffic data?
iii) How will the fuzzy logic based architecture be designed?
iv) What level of complexity is allowed for the algorithm?
v) What level of efficiency and availability is expected for the proposed system?

1.5 Justification

The system will be beneficial to the following:

The Government. In that by adopting an adaptive traffic control system, they will be able to effectively reduce traffic congestion levels. This in turn will have a positive impact on the economy which is currently affected substantially by traffic congestion. With decreased delays on the roads there will be an increase in worker productivity rates which will in turn boost trade opportunities.

The Commuters. The system will reduce the overall waiting time. This is because the system will be allocating green time based on priority i.e. which lane needs green light because it has waited the longest or which lane needs more green light duration because it has more vehicles queued up. The commuters will be able to reach their destinations much faster as well as save on fuel consumption.

The Traffic Policemen. With this system in place, the policemen are saved from standing for long hours a day controlling traffic. The system will run autonomously and make intelligent decisions on behalf of the traffic personnel. Their presence on the roads will not be needed most of the time; unless to enforce traffic rules for instance; in the event where road users fail to obey the traffic light signals.

1.6 Knowledge Contributions

While this thesis adopts similar methods used in related works, the main contributions are:

1. *We introduce a unique intelligent hybrid approach that integrates a WSN and an FLC to control traffic lights.*

The WSN is able to collect and store large amounts of environmental data at minimal costs and the FLC will use this data in its knowledgebase to derive traffic routing decisions.

2. *We integrate a unique algorithm that autonomously routes traffic based on FLC processed data.*

The algorithm uses the output variable (PD) processed by the FLC to dynamically control the traffic lights at the junction based on urgency. That is, it assigns green light to the lane with the highest PD value.

3. *We propose a new way of how the weaknesses accrued by the lack of learning ability of fuzzy systems can be solved without having to integrate an artificial neural network.*

The proposed approach incorporates a mini program that stores the traffic data collected from the WSN. This program can then help experts to review and analyse the data to come up with traffic patterns over a specified period of time. This information can be used later to modify /adapt the model to the derived patterns. For instance if from the analysis the experts realise that at a certain time of the day lane 1 always gets the highest number of cars or lane 3 usually waits the longest for green light then they can modify the system according to that pattern for that specific period only. Also in the event the WSN experiences a downtime, the traffic lights can be controlled using the stored data in the mini program/reference log file.

1.7 Scope

This research is limited to and concerned with controlling traffic lights at four or less approach junctions. The limitations of the model are:

i) It does not include the synchronization of traffic lights among the different intersections, such that, intersection one operates independently of

intersection two decisions. The system assumes that the opposite road to which a lane flows into is free/not occupied.

ii) It does not control traffic at an intersection with more than the four approaches (North, East, South and West).

1.8 Outline of the Thesis

The rest of the thesis is organized as follows:

Chapter 2

It discusses what has already been done to address the road traffic congestion problem. It provides an in-depth review on the techniques that are currently in use to manage road traffic and points out the motivations that influenced use of WSN and FLC in this research.

Chapter 3

This describes in detail the methods as well as development tools and procedures that were employed during this research.

Chapter 4

This gives a detailed account of how the proposed approach was evaluated in a simulated environment. The chapter further discusses the research findings of the experiments carried out to illustrate efficiency of the proposed approach.

Chapter 5

Concluding remarks and recommendations for future work are given in this chapter.

CHAPTER TWO
LITERATURE REVIEW

The steady increase in number of vehicles on the roads has caused an increase in traffic congestion in most urban cities around the world. One measure being taken to solve the congestion problem is the expansion of roads. However, infrastructure investments alone will not help eliminate traffic congestion. Downs (1962) confirms that an extension of roads is met with a proportional increase in traffic. The negative effects of traffic congestion have prompted scholars to further investigate and come up with other solutions to address this traffic jam problem.

This review serves to discuss what has already been done to address this traffic jam problem. It discusses the techniques that are currently being used to manage road traffic. In addition we describe some of the benefits and limitations of these solutions. The chapter will be organized into four main topics; Sensing Systems, Existing Traffic Control Systems, Research gap and Conclusion.

2.1 Sensing systems

2.1.1 Sensor Networks

Advancement in very large scale integration (VLSI) and semiconductor technologies have enabled the development of smaller, tiny, low power, and inexpensive sensors and controllers/microprocessors. Furthermore, developments in wireless technologies have made it possible for the use of sensors to collect large amounts of environmental data at minimal costs. In light of this, implementation of distributed sensing systems proves to be one of the sustainable solutions to road congestion. This is because sensing systems provide a cheaper alternative to that of expansion of roadways and building of new infrastructure. (Dargie *et al*. & Poellabauer, 2010)

Sensor networks have been used to gather real time information about the density, sizes, and speed of vehicles on the roads. For instance, Knaian (2000) proposed the adoption of sensors to manage traffic flow in Vassar Street, Cambridge, Massachusetts. He deployed a prototype that used magnetic sensors. Each sensor node consisted of two Anisotropic Magneto-Resistive (AMR) magnetic sensors, one at the back and another placed in front of the node (Knaian, 2000). The movement and speed of a vehicle is captured by observing the disturbance it creates in the Earth's magnetic field. To determine movement, as the vehicle approaches the sensor it pulls field lines away from the sensor and then towards the sensor when it drives away from it; to measure speed, once the node detects a signal from the rear sensor's baseline it begins to count number of samples until the signal from the forward sensor crossed the baseline (Dargie & Poellabauer, 2010).

Developing countries are also moving towards this approach. Kenya has since adopted the use of these sensing systems. On 8th October 2013, IBM opened a commercial technology research facility based in Catholic University of East Africa in the Karen suburb. The lab's research agenda includes the development of cognitive computing technologies which integrate learning and reasoning capabilities enabling experts to make better decisions in the search for solutions to Africa's most pressing challenges (IBM, 2013). In an effort to tackle the traffic congestion problem, IBM partnered with Kenyan internet service provider Access Kenya to develop a pilot solution to enable Nairobi commuters to use their mobile phones to get advice on driving routes through the city depending on estimates of traffic congestion.

The project dubbed Twende Twende, is a mobile application that uses specialized algorithms to do image processing and interpret visual data received from closed-circuit television (CCTV) cameras positioned around Nairobi. Motorists are then able to get information on a) what areas to avoid because of congestion by suggesting alternative routes and b) updates on road conditions to allow them get from point A to point B safely. This information is retrieved via an SMS-based query for basic phones and on smart phones the service is accessed via an application through which users can view a map of the city showing route options and potential traffic hotspots.

The project's main focus is data collection, analysis and information dissemination, it does not actively control traffic on the roads. (IBM, 2013)

2.1.2 Wireless Sensor Networks (WSN)

When sensing systems are comprised of hundreds or even thousands of nodes, it is nearly impossible to wire all these distributed sensors. Therefore wireless technologies (such as Radio Frequency, Infrared and Laser) are incorporated, resulting in a wireless sensor network (WSN). Wireless sensors not only communicate with the base station but also with each other. For this reason, the nodes in the WSN often assume additional responsibilities, such as, own processing, communication and storage capabilities, as opposed to relying entirely on the base station (BS). (Dargie & Poellabauer, 2010)

Owing to the fact that sensors are microelectronic devices, they operate on a limited energy budget. For this reason, WSNs need to regulate their energy consumption during communication. One common route used is multi-hop communication (mesh topology), whereby, large distances between the nodes are split up into several distances. The sensor nodes then serve as relays for other sensor nodes and find the most energy efficient route to transmit sensor data towards the BS. Short range transmission minimizes power consumption during data transmission. Due to the wide geographical coverage of sensor networks, multi-hop is preferred because it conserves more energy as opposed to the single hop (star topology) where all nodes directly communicate with the base station regardless of the distance (Dargie & Poellabauer, 2010).

WSNs in road traffic monitoring have recently become prominent. This is because of their ad hoc and easy deployment capability. Large numbers of sensors can easily be installed throughout the road network. They are also flexible. The network can be expanded without interfering with existing network infrastructure. Pascale *et al* (2012), introduce an ITS that uses WSN to monitor traffic. Their system comprises of a network of traffic sensors deployed throughout the roads that collect and forward measurements to a remote server. The server aggregates and processes macro-

parameters of traffic flows arising from heterogeneous monitoring systems, then it distributes the data to traffic management centres, road control units (RCU) and information providers. The macro-parameters can be used for traffic analysis and management and road safety applications. (Pascale *et al*, 2012)

Zhou *et al* (2013) further back up Pascale *et al* that efficient traffic information collection is an important part of traffic monitoring systems. In their paper they introduce a framework that uses WSN to collect traffic information based on user requests. Three types of sensors are deployed for data acquisition. They further propose a data centric routing scheme for traffic information delivery to the user based on the user's required traffic information. (Zhou, *et al*, 2013)

The two mentioned frameworks successfully prove that use of WSN to monitor and collect real time road traffic data is an optimal solution, especially due to the uncertain nature of traffic. WSN offer a high level of adaptability to managing these ever changing road traffic characteristic. These systems collate information; infer congestions; and suggest to drivers some alternative routes and emergency exits (Dargie & Poellabauer, 2010). However these discussed systems do not actively control traffic – their main focus is data collection, analysis and information dissemination.

2.2 Existing Traffic Control Systems

Almost all urban cities in the world use traffic lights to control the traffic on the roads. The traffic lights switch from red which means stop to green meaning move. Over time there have been developments of different types of traffic light control systems: static systems and smart sensor based systems; each with their own advantages and disadvantages.

2.2.1 Static systems

These systems use fixed plans. The traffic lights timing and switching patterns are predetermined despite prevailing traffic conditions. Such systems are commonly used in emerging economies because of low costs of implementation. They do not

require advanced communication systems to be installed in vehicles such as sensors or radio frequency identification reader (RFID) tags. Consequently this lack of intelligent and/or adaptive strategies in these devices does very little in improving the road network performance and the traffic congestion levels. One of the major downside of this approach is that the lights do not operate with real time traffic data. This means that they do not take into account the non-uniform nature of traffic. As a result, it does not matter whether at a particular period of time route one has more cars that route two; the green light allocation time still remains the same for all routes possibly causing increased traffic on route one (Kumar, 2011).

For instance Runyoro *et al*. (2014) provide evidence of three cities in Tanzania that are greatly congested due to use of old technology static traffic lights. They further propose a framework to assist in automation of road traffic management. Moreover, Kenya also experienced the shortcomings of these static lights in early 2014 when the county government decided to pull out traffic personnel and let the lights control traffic. This was so that they can test the efficacy of the lights. However, the demonstration was a failure which resulted in a four hour bumper to bumper gridlock (Ndonga, 2014).

India tried to use VA lights but this approach also is ineffective especially when controlling traffic at major intersections (Kumar, 2011). VA lights switch to a green signal when they detect presence of cars, however, they do not count number of cars present (Fahmy, 2007). As a result, more than one approach can have a green signal for as long as the VA detector registers a car's presence despite the traffic density of cars in the individual approaches. This will consequently lead to traffic congestion and confusion, with simultaneous movement of cars from different directions of the junction.

2.2.2 Sensor based smart systems

The development of smart traffic light control system was as a result of scholars trying to solve the weaknesses of static traffic light systems. Smart traffic light controls are dynamic. This means that they use real time data to make priority based

decisions. They use advanced communication systems based on sensors and/or RFID tags to collect data and provide the system with information on current situation on the roads (such as number of vehicles on individual roads or how long vehicles have been waiting for green light). The smart system then uses an algorithm to process this information and make decisions; that is, it automatically determines the duration of each traffic light signal based on prevailing traffic situation on the roads. Commonly used systems include fuzzy expert systems, artificial neural networks and wireless sensor networks.

a) Fuzzy Expert Systems (FES)

Fuzzy expert systems are a suitable approach to dynamic traffic signal control because of the nature of uncertainties on road traffic where the traffic distributions fluctuate non-uniformly. Fuzzy logic is a field started by Zadeh (1965). It is a superset of Boolean logic that has been extended to handle partial truths between completely false (0) and completely true (1). This is in an attempt to mimic or reflect how humans think, to model our sense of words when describing certain phenomena as well as our common sense in decision making. The sensors collect data from the environment which in turn is fed into the fuzzy controller for processing. The inference process in a fuzzy controller is similar to the way a traffic policeman handles the traffic flow at a typical roundabout. It assigns green or red light signal based on urgency or as traffic fluctuates; and selects the best decision that will minimize congestion at a particular interval. (Fahmy, 2007; Khiang, Khalid, & Yusof, 1997., para. 5)

Khiang *et al* (1997) present a fuzzy logic traffic light controller. Their approach uses two input variables; quantity of traffic on the arrival side (arrival) and quantity of traffic on the queuing side (queue) collected from the sensors on the lanes. Their system controls traffic on multiple lanes simultaneously i.e. North and south lanes move together while east and west lanes move together. When North and South have green light, East and West stop (queue). The fuzzy controller observes the density of north and south as one side and east and west as another side. Their system then determines green light allocation and extension based on the side that has the highest

traffic quantity. From their experiments, they are able to demonstrate that their fuzzy logic traffic light controller performs better than the fixed-time (static) controller.

From a comparison made between the performance of the fuzzy logic controller and that of a fixed-time (static) controller; Khiang (1997) observed from the results that the fuzzy logic controller had a lower average waiting time – a difference of 6 minutes.

Fahmy (2007) later presents another system, fuzzy logic adaptive traffic signalling control (FLATSC) that uses fuzzy logic controller to manage traffic at a four intersection roundabout. However, unlike Khiang *et al* (1997) system, his system employs another input variable, waiting time, to determine green light allocation and extension. FLATSC therefore uses traffic quantity and waiting time to determine the priority degree (output variable) for each lane on the roundabout. The output value is the green light time/extension for each lane. The lane with the highest output value gets allocated the green light. When cars in one lane move the other lanes stop. The green light extension was not a fixed value; it was dependent on real time data collected from the sensors. The value changed as the traffic variables fluctuated from cycle to cycle and/or lane to lane. This ensured that traffic was controlled based on prevailing traffic conditions on the roads.

From a comparison with the fixed controller and vehicle actuated systems, FLATSC proved to be more effective in managing the changing traffic patterns. In addition FLATSC attempted to resolve starvation. Starvation describes a situation whereby some lanes end up always getting last priority because they usually get the least traffic consequently ending up always queuing for the longest time during a cycle. FLATSC addressed this issue by incorporating waiting time as a factor in determining green light allocation. For instance a lane that has low traffic but very high waiting time still has a chance of getting a high priority degree just as a lane with high traffic but low waiting time; depending on the fuzzy inference.

Although use of fuzzy expert systems in traffic light control systems enhances the efficiency of traffic movement in roads, the downside of such systems is that, they do not have the ability to learn, they are not adaptive. Meaning, they do not incorporate past knowledge or experiences to make current decisions; rather they only make decisions based on the current knowledge they have of the situation.

b) Artificial Neural Networks (ANNs)

The major difference between ANN (learning systems) and FES is that; while an FES uses present knowledge to make decisions, in a learning system, the decisions are computed using the accumulated experience or knowledge from successfully solved examples. Since ANNs try to mimic the human brain they possess an adaptive feature that allows each node within the network to modify its state in response to past and present knowledge. (Beattie, 2011; Patel & Ranganathan, 2001)

An adaptive traffic light system was modelled using ANN approach. Dai *et al* (2011), designed a neural network (NN) based signal controller to control the traffic lights in an urban traffic road network. The input given to the ANN models are the list of data collected by the sensors which are placed around the traffic lights. The sensors give the traffic light ANN model all the data which are related to the past and present traffic parameters. The model then processes this input and selects the most suitable output that suits current traffic situation. These results are then used by the traffic lights to set the timing for the red and green lights (Mittal & Singh, 2013).

Michael *et al*. (2014) also present a neural networks based traffic light controller called Environment Observation Method based on Artificial Neural Networks Controller (EOM-ANN) to control urban traffic. Their approach is different from (Patel *et al*, 2001; Dai *et al*, 2011) because they also incorporate mathematical strategies (EOM) to make signal allocation decisions. EOM is a mathematical methodology for obtaining timing plans for isolated intersections. It achieves this by calculating the minimal green time for each phase then to prevent congestion, an additional green time is allocated to each lane that still has cars even after getting green light. However, the downside of using EOM and most traditional mathematical

models in traffic control is that it sets traffic light timing based on averages of the basic parameters (Dai *et al*., 2011). Due to the fact that these basic parameter figures are constants, the EOM does not incorporate the real time nature of traffic that is characterized by the constant change in traffic parameter values. The EOM-ANN is an attempt to resolve this issue of real time data, Michael *et al*., propose use of ANN to obtain this traffic data patterns. That way, the green light timing and allocation is based on actual/prevailing traffic conditions rather than analytical calculations. (Michael *et al*., 2014)

EOM-ANN uses the feed-forward method with 8 neural nodes in total for input, hidden and output layers. It is further divided into two modules; reviser and the neural. The former defines correct traffic light timing and the latter provides the most appropriate value for the current traffic behaviour. The inputs of the ANN are the number of light, medium and heavy vehicles. (Michael *et al*., 2014)

From a comparison between static time controller, EOM and EOM-ANN EOM-ANN reported better traffic flow and congestion management. The average traffic flow of the individual controllers was as follows: static controller - 82.55, EOM - 68.70 and EOM-ANN registered an average of 53.75. (Michael *et al*., 2014)

While Dai *et al*. (2011), Michael *et al*. (2014) and Patel *et al*. (2001) proved in their work that using ANN to manage traffic was effective; ANNs training process in most cases is a time-consuming task requiring the application of input training patterns in an iterative manner. This was experimentally demonstrated by Barbosa *et al* (2010), who showed that by increasing the amount of data the performance of the ANN system improved. The error margin was lower when training was extended to 15 minutes of collecting data as opposed to 5 minutes.

Also another drawback of ANNs is the lack of rules or guides to support the decisions to be made; resulting in development of solutions that are mostly specific or case base problems. This means no explanation or guarantee that the solution chosen is the optimum one (Barbosa & Pinto, 2010). This is proven by Patel *et al*

(2001) who demonstrate that the ANN had a correct decision rate of 73% as opposed to the intelligent decision making system for urban traffic control applications (IDUTC) and FES which both had a 95% rating. They further realised in their experiments that the ANN approach had difficulty in generalizing on the various numbers and the combinations of traffic parameters and required cycle-time adjustments (desired outputs). (Patel *et al*., 2001)

c) Hybrid systems

To overcome the limitations of the individual implementations mentioned above (ANN and FES approaches), such as; lack of learning ability of fuzzy systems and high implementation time costs as well as lack of inference process of ANNs, Patel *et al*. (2001), developed IDUTC. This approach integrated the learning abilities of an ANN and the knowledge-based decision-making ability of the FES. The back propagation-based ANN allowed the model to learn and adapt to the dynamically changing environment and the FES was employed for decision making using the IF-THEN rules.

A summary of the simulations for the IDUTC, the ANN, and the FES approaches indicate that: the IDUTC provided 95% correct decision rate and an average waiting time of 2.186 minutes. It relieved intersection congestion better than the ANN approach which provided 73% correct decision rate and an average waiting time of 2.958 minutes. While the FES approach correct decision rate was equal to that of IDUTC, it was observed that the computed decision did not lead to a better reduction in the wait times. The FES had an average wait time of 2.975 minutes which is lower than the other two approaches. (Patel *et al*., 2001).

However, the initial implementation cost of hybrid systems can be quite costly, considering that A.I is still considerably a new research area in traffic control. Also the fact that a hybrid is a combination of two or more smart traffic control system (STCS), the development time can take a long time in an attempt to ensure successful integration of the different systems.

d) WSN-based traffic control systems

In the event WSN is used to not only collect traffic data but also actively control road traffic, additional functionalities are incorporated into the network's controller. An algorithm is embedded to control the traffic lights – it generates routing decisions based on sensor data aggregated.

Yousef *et al*. (2010), present an adaptive traffic light control model for single and multiple intersections using WSN. Their work uses the WSN to route traffic based on traffic density and waiting times. It is composed of: sensors that detect the presence of vehicles and have a memory that stores their waiting times on each road. It also has an intelligent traffic controller that processes the sensor data then employs three algorithms traffic system communication algorithm (TSCA), traffic signal time manipulation algorithm (TSTMA) to route traffic based on the traffic variations of all lanes of the intersections at a particular time and traffic control algorithm on multiple intersections (TCAMI).

TSCA main objective is to enable exchange of information between the sensors' base station (BS) and the controller using a direct routing scheme approach. This means all sensors are within range of the BS and directly communicate with it. On the other hand, TSTMA main responsibility is to set the traffic signal duration in an efficient and dynamic manner such that traffic flow is maximized while at the same time ensuring minimal average queue length (AQL) and average waiting time (AWT). TSTMA makes use of the traffic information gathered at the traffic BS from the sensors to calculate in intelligent manner, the expected queue length, for the next traffic cycle, and then schedule efficient time setting for the various traffic signals. TSTMA achieves this objective through three main techniques: (a) Dynamic selection and ordering of the traffic phases based on the number of lanes allowed in the intersection; (b) Dynamic adaptation to the changes in the arrival and departure rates and thus dynamic decisions about queues' lengths and their importance; (c) Dynamic control of the traffic cycle timing of the green and red periods. TCAMI main objective is coordination and setting of traffic parameters and conditions on the multiple intersections in general and on the successive intersections in specific, with

the objective of minimizing delays, caused by stopping, waiting and then speeding up during road trips (also known as green wave – where drivers need not stop on multiple intersections thus achieving, if implemented correctly, an open route for the vehicles). When TCAMI is executed on each intersection it will generate traffic information, which in turn represents an input to the subsequent intersection, and so on. As such, the traffic flow will be controlled in a flexible manner. (Yousef *et al*., 2010)

Algorithm 2: *Traffic Signal Time Manipulation Algorithm Description*

Input: $D^i, N, M, \overline{\mu}^i, \overline{\lambda}^i, T$.

Output: AWT^i, AQL^i.

Operations:

1. Generate a truth table for all possible combinations of the directions based on D^i. Then, generate another truth table using the Conflict Directions Matrix to generate all safe combinations of directions by canceling the combinations which include any traffic conflict in the first truth table.
2. Define the traffic array to hold all the safe combinations up to *r* rows, the queue length for each D^i, and the green and red periods.
3. Initialize all queues' lengths to zero, the green period to $T/4$ and red periods to $(T-(T/4))$ for all active directions.
4. Compute the queue for each active direction *i* as follows:
   - (Inter-arrival time) $t_a = -\ln(U)/\overline{\lambda}^i$ for every instance *a* of the *T* period.
   - (Inter-service time) $s_a = -\ln(U)/\overline{\mu}^i$ for every instance *a* of the green period.
   - Average inter-arrival time for active direction *i* and average inter-service time respectively as follows:

     $$\overline{t}_i = \left[\sum_{a=1}^{T} t_a\right]/T \quad , \quad \overline{s}_i = \left[\sum_{a=1}^{G^i} s_a\right]/T$$

   - $\lambda^i = 1/\overline{t}_i \qquad \mu^i = 1/\overline{s}_i \qquad \therefore \rho^i = \lambda^i/\mu^i \qquad Q_j^i = (\rho^i)^2/1-\rho^i$
5. From the traffic array, the queues' lengths for all the directions is summed after updating the queues' lengths from the previous step to find the number of phases (safe combinations). So, this step involves:
   - Computing $\sum_{i=1}^{12} Q_j^i$, for j =1 to *r* (safe configurations).
   - Selecting the phase with the largest sum as the first phase. The next largest sum becomes the second phase, and so on. The process repeats until all directions are found in the safe array and fit into a maximum of four phases.
6. Use the safe array and the maximum values of the four phases, to find the green and red times for each phase.
   - $G_i^k = \dfrac{\max_k}{\max_1 + \max_2 + \max_3 + \max_4} \times T$, $\qquad R_i^k = T - G_i^k$
7. Compute the expected queue length and waiting time for the next cycle for each active direction and update all the directions queues' lengths
   - $Q_j^i = Q_{j-1}^i + \lambda^i G_i^k - \mu^i G_i^k + \lambda^i R_i^k$, $j \in (1 \ldots T)$: represents cycle #
   - $\overline{WT}_j^i = Q_j^i / \lambda^i$
8. Check if the current number of cycles matches the simulation cycles entered by the user,
   - if yes *goto step 12*. if($j == N$) then *goto step 13*
9. Check if the current number of cycles matches the cycles entered by the user to regenerate new arrival and departure rates,
   - if yes go to step 11 else *goto step 5*
   - if($j == M$) then *goto step 11* else *goto step 5*
10. Input from the user to enter the arrival and departure rates parameters for the Poisson generation for each active direction $(\overline{\mu}^i, \overline{\lambda}^i)$
11. Go to step 5.
12. Calculate the average queue length and the average waiting time for each active direction.
    - $AQL^i = \left[\sum_j^N Q_j^i\right]/N$
    - $A\overline{WT}^i = \left[\sum_j^N \overline{WT}_j^i\right]/N$

Figure 0.1: High level description of traffic time manipulation algorithm (TSTMA)[Yousef *et al.*, 2010]

To show efficiency of proposed scheme, Yousef *et al* compared the system to the traditional traffic light control approach which uses static plans i.e. fixed time control. The results indicate that the proposed system had a better performance rate in managing traffic; its AWT was much lower at 2.98 minutes compared to 7.87 minutes of the fixed time controller. A low AWT means that the flow of traffic is

increased hence lower average queue time (AQT) of 9 cars as opposed to 36 cars per queue in the fixed time controller. The dynamic approach was able to handle queues quickly with less cars accumulating on a lane during the observed time (Yousef *et al*., 2010).

Bhuvaneswari *et al* (2012) further support Yousef *et al* by developing a traffic congestion control system – adaptive traffic signal flow control using wireless sensor networks (ATSWSN) that is adaptive in nature and provides time slots to each route based on traffic density. The system collects real time data using infrared (IR) sensors and the microcontroller's scheduled algorithm that performs all intersection control traffic light functions based on the data fed from the sensor nodes. When compared to the conventional fixed time approach, ATSWSN registers a higher traffic flow rate and as a result lower average waiting time. They observed that in the fixed time approach, the waiting time increased as the number of vehicles increased irrespective of their speeds and speed factors. (Bhunaneswari *et al*., 2012)

The mentioned approaches have one limitation; they do not take into account 'starvation'. They control the light signals based on traffic density only. For this reason, it is very likely that a particular lane that usually has low traffic levels will always be given last priority or might not get green light altogether; consequently always registering higher AWT during the signal cycles/phases. A downside of this is that, motorists will avoid this lane and move to the lanes with lower AWT. This will lead to an increase in congestion levels on the lanes that usually get first priority.

The downside of using this approach is accrued to the fact that sensors are micro-electric devices, this means they operate on a limited energy budget. For this reason WSNs are faced with the problem of having to regulate their energy consumption. This can be a daunting task especially when dealing with large complex traffic networks with multiple intersections. The interdependency of each intersection on its neighbours makes it extremely important to ensure that the different sensors in each intersection are in constant communication to ensure real time data processing. This consequently takes a toll on the energy consumption rate of the sensors especially if the distance between them is wide. The wider the distance between nodes and BS the

higher the attenuation rate; consequently leading to increased power needed during data transmission/communication. Failure of a node could lead to massive traffic congestion; also, a downside of this is that motorists will avoid this lane and move to the lanes with lower AWT. This could lead to an increase in congestion levels on the lanes that usually get first priority.

2.3 Research gap

From the extensive background research on previously done work, the following are the gaps we identified:

- Most hybrid systems only incorporate intelligent systems; fuzzy systems and neural networks, to control and manage traffic.
- Most WSN based approaches do not take into account issue of starvation. They control the traffic lights based on traffic density only, the waiting time or queuing time of the cars is excluded in the algorithms.

2.4 Conclusion

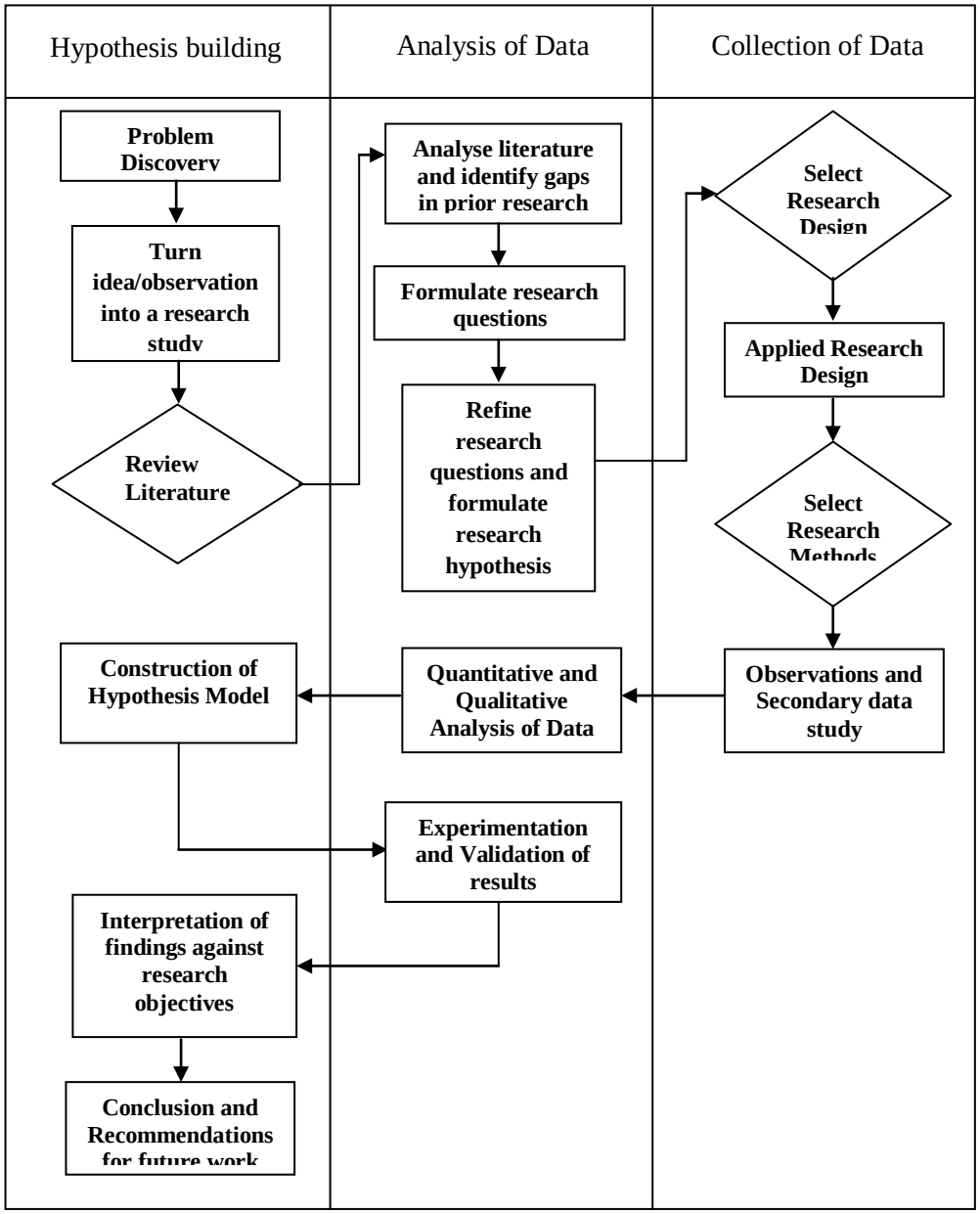
A review of the several investigations carried out on the use of technology to control and manage traffic was presented in this chapter. It is observed that the implementation of information communication and technology (ICT) on transportation systems does have a major impact on traffic levels. While the static systems provide a simpler method of automatically controlling traffic; they do not have the flexibility needed on most urban junctions which serve non uniform traffic from the various approaches/roads. Advancements in AI further led to the development of intelligent traffic control systems. The main objective of these smart systems is to have the traffic lights mimic the human intelligence thus eliminating the need of having policemen control traffic on the roads. These intelligent systems provide a way for the lights to change from red to green based on current traffic conditions. Though these systems provide substantial benefits to management of traffic, fuzzy expert systems and ANNs are a branch of AI which is still an emerging field in information technology (IT); hence the implementation of such systems as stand alone is still quite costly, especially in the developing countries. An alternative to using the intelligent systems is sensor networks. These networks gained popularity

especially due to the low cost of implementation compared to the AI based systems. The network is comprised of many sensors that cooperate to monitor and collect data about traffic conditions on the roads. This information is then forwarded to a controller that processes the data into meaningful information. Using an algorithm the controller is able to make routing decisions based on current traffic conditions.

CHAPTER THREE

METHODOLOGY

This section, discusses the methods/techniques as well as development tools that were employed while carrying out the research. The first section describes the research methodology – where it discusses the type of research design, research methods and data collection tools used to gather information during the research. Second section describes the software development process – where the techniques that were used to demonstrate our proposed approach are discussed. Fig. 3.1 is a summary of how this research was carried out.



3.1 Research Methodo

Figure 0.1: Research Process Flowchart

3.1.1 Research Design

Given the practical nature of the thesis, applied research design was adopted. This design deals with gathering information related to a specific subject; then using that information to determine what worked and what did not work so as to come up with a better way of solving a practical real life problem. This design was convenient for this research since the main aim was to demonstrate the efficacy of using smart traffic control systems i.e. WSN and FLC to route dynamic traffic. Applied research design methods, made it possible to gather necessary information, analyse it and develop a novel approach/solution to the traffic jam problem.

3.1.2 Research Methods/Process

The works reviewed in this thesis were selected and analysed based on the following criteria (Hawi *et al*, 2015):

i) Approaches used to make traffic routing and light signal allocation decisions. For instance adaptive (learning) versus non-adaptive strategies; offline versus real time strategies; and hybrid strategies.

ii) Number and types of parameters/variables (input and output) used. We review systems that use single variables (e.g. traffic quantity) and ones that use several variables (e.g. traffic quantity, waiting time, past and present traffic data knowledge) to make traffic routing decisions.

iii) Traffic data collection methods used (such as sensor types) and communication methods applied (such as multi-hop or single-hop) to transmit collected data.

iv) STCS that control traffic at an isolated junction or multiple intersection junction or both.

v) Ways to improve overall performance of already existing intelligent/smart systems (STCS) in use.

3.1.3 Data Collection Tools

The techniques used to collect data include:

i) Studying existing literature

This research was mostly built upon the course work done and elements of past research. This formed a rich source of information and was the basis of all the other data collection techniques to be used. It allowed for a thorough review of the existing related work done – the accumulated theories, knowledge, methods, and techniques on the research variables; traffic, WSN and traffic lights. The bodies of knowledge used as references include: studying journals, online articles, reports and technical manuals.

To provide for a good grasp of the study, it was necessary to conduct an extensive literature review and background study on topics such as; WSN for dynamically and automatically controlling traffic lights, intelligent decision making systems for urban traffic control using ANNs and fuzzy expert systems and Traffic congestion in urban cities. This tool was effective in helping to identify existing gaps – through evaluation of what worked and what did not work previously.

ii) Observations

This tool was inevitably employed in daily commuting activities. Commuting to and from home, enabled firsthand experience and collection of data on the effects of traffic congestion; such as how many hours on average are spent stuck in traffic daily or how effective the static traffic lights systems are in controlling the dynamic nature of traffic on the roads throughout the day. This tool was particularly effective when answering the following research question; 'How does the current Nairobi vehicle traffic management system operate at junctions?'

3.2 Development Tools

3.2.1 Process Model

Prototyping

In order to effectively control the design of the thesis experiments, demonstrate as well as evaluate efficiency of the proposed approach, we developed a model of a smart traffic light control. This model is known as the prototype. It enabled us to build an experimental prototype for our approach rapidly. It also allowed us to have an early look of the proposed approach through an iterative experimentation process. Moreover, it allowed previous sections to be revisited if new designs were realized. This made it easier to review the performance of the model, and detect any deficiencies during the development process (Munassar & Govardhan, 2010). While the prototype has its benefits, the structure of the research can be compromised due to constant changes. Therefore, careful consideration was taken when managing scope change.

3.2.2 Development Procedure

To demonstrate the proposed approach, this thesis developed a fuzzy logic based software system based on Java and qtFuzzylite 4.0

3.2.2.1 Conceptual Framework

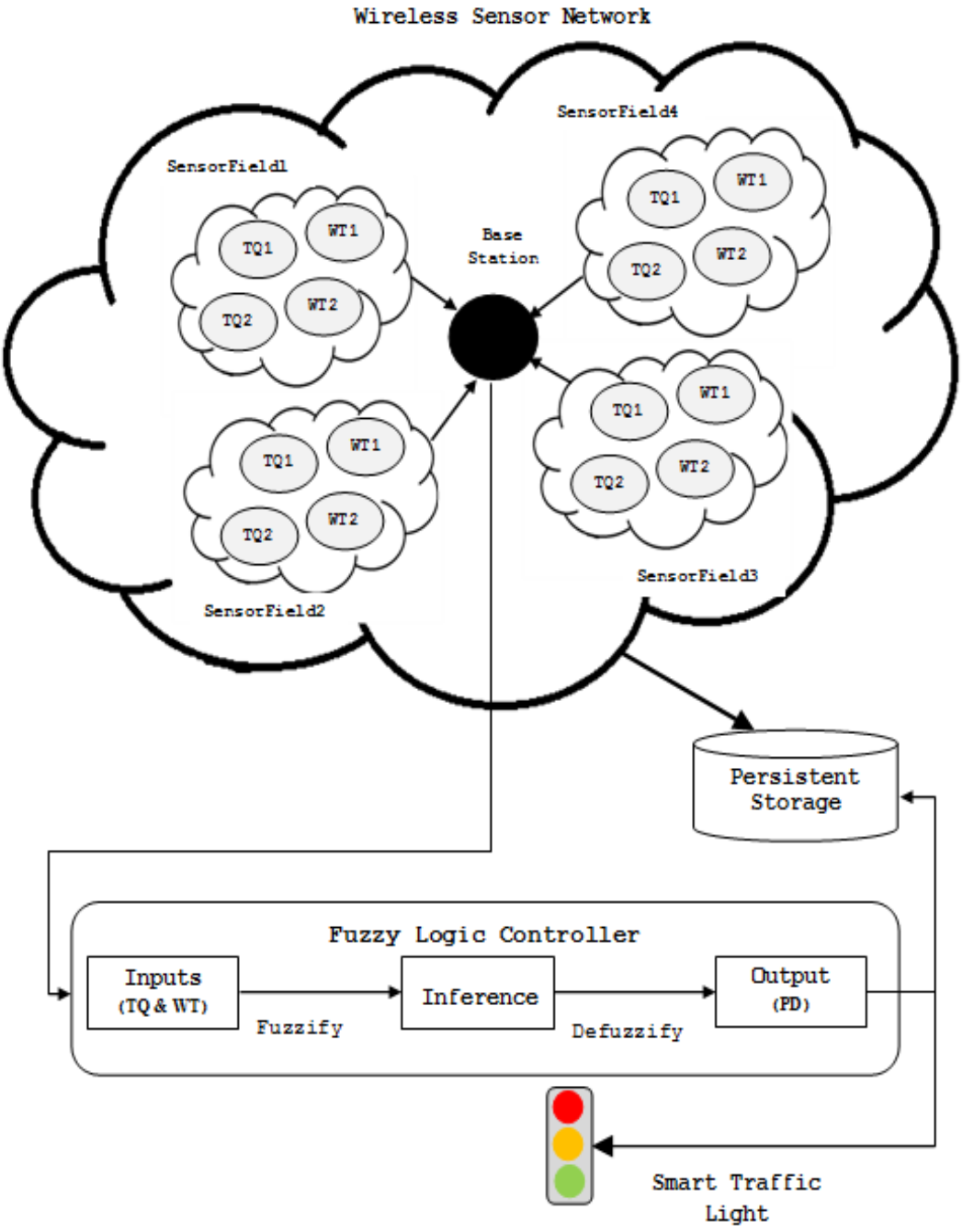


Figure 0.2: Conceptual Framework for Smart Traffic Light Control Using Fuzzy Logic and WSN

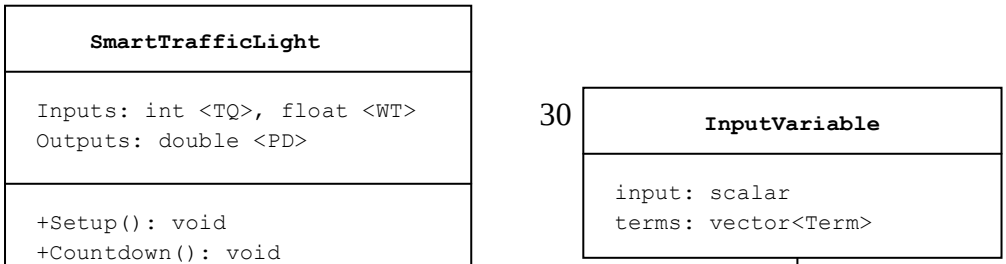
3.2.2.2 Design of the Software System

Figure 3.3 describes the abstract class diagram of our software system. Brief Description of the class diagram's components:

SmartTrafficLight – This is the overall class that holds all other components. It provides the interface for the entire system. It has an *aggregation relationship* with TrafficLightController. This means the two classes are peers and can exist independently of each other. TrafficLightController can only have one instance of SmartTrafficLight ; whereas, SmartTrafficLight can have more than one instance of TrafficLightController. *Setup()* and *StartingCar()* are responsible for creating the system's graphical interface; *Countdown()* is responsible for comparing the PD values (derived from MyListener class) of all four lanes so as to determine which lane gets assigned green light first; *CalculateTicker()* is responsible for calculating the green light duration of each lane based on inputs (TQ and WT derived from MyListener class); *Repeat()* is responsible for looping all the processes involved in controlling the traffic lights, so as to ensure program runs continuously without human input. Lastly *main()* ensures that the program runs, without it, all other operations will not function.

MyListener – this is an inner class of the SmartTrafficLight class. This is denoted by the circle with a plus sign in figure xxx. It's *actionPerformed()* method is responsible for carrying out the entire fuzzification process; i.e. handles all the processing from inputs to inference to defuzzification and finally output. As a result, *MyListener* has a *dependency relationship* with TrafficLightController class. This means that if the specifications of the TrafficLightController change then MyListener will also have to be updated accordingly. This relationship is denoted in Figure 3.3 by the broken arrow.

TrafficLightController – this is the fuzzy logic controller engine. Every time the inputs are set *process()* is called. This class has *composition relationship* with three other objects, namely; *InputVariable*, *RuleBlock* and *OutputVariable*. This relationship is denoted by the black (filled) diamond symbol in Figure 3.3. *FireRules()* in RuleBlock is responsible for triggering the inference process. The *defuzzify()* in OutputVariable is responsible for defuzzifying the accumulated outputs into crisp set values.



3.2.2.3 Description of the Software System

i) Java

We developed a model of a smart traffic light control supported by fuzzy logic and WSN using java language. This enabled the simulation of the proposed approach in a virtual environment. Consequently this provided the research with an early look at the functions and interactions between the WSN and FLC through an iterative experimentation process.

- Assumptions of Simulation Setup

The following were the assumptions considered during simulation setup (implementation).

1. The roundabout is an isolated four-way junction with traffic coming from west (lane 1), south (lane 2), east (lane 3) and north (lane 4). It follows the left hand drive rules.
2. The Base Station of the WSN has already aggregated the sensor data from all the 32 sensors. Therefore, our system only uses the two inputs TQ_T and WT_T derived from aggregation (refer to equation 1 and 2 below).
3. Only one approach gets green light signal at a time. There is no concurrent movement of vehicles from different approaches.
4. The minimum and maximum number of cars any lane can get at a time is 0 cars and 30 cars respectively; while the maximum duration of green light any lane gets is 3 minutes (see equation 4).

The following are the components that were directly created in the Java IDE:

a) WSN

Prior to describing the simulated WSN, it is necessary to briefly describe the actual WSN that enabled us to create the simulated WSN version (that had its basis from the second assumption in section 3.2.2.2).

❖ Actual WSN

1. *Sensors:* 32 in total. Each approach at the junction has 4 sensors, 2 that collect the number of cars (TQ1, TQ2) and two collect the time the cars

in a lane wait for a green light signal (WT1, WT2). TQ1 and WT1 are placed at the entrance of the lane so as to; count the cars entering the lane for the current cycle and record the time the cars of current cycle entered the lane respectively. WT2 and TQ2 are placed at the exit to record the time cars in the previous cycle crossed the light and record number of cars that cross the lane in the current cycle respectively.

2. *Base Station (BS)*: responsible for aggregating the data from the sensors and prepares it for processing by the FLC embedded in a remote computer. The BS compares data collected from the 4 sensors and analyses them. The difference between the respective sensors gives the total number of cars queued up (TQ_T) and the total amount of time waited for green light (WT_T). Owing to the fact that the system is designed for an isolated junction; there is no communication between sensors of two or more junctions. As a result, multi-hop communication was not necessary. Single hop communication proved effective due to the short range communication between the sensors and the BS. Below are simple mathematical equations describing how the BS aggregates and analyses the sensor data:

$$TQ_T = TQ2 - TQ1 \ldots (1)$$

$$WT_T = WT2 - WT1 \ldots (2)$$

3. *Persistent Storage*: this is where the aggregated sensor data, TQ_T and WT_T are stored in a file. They can later be used by experts in the event they need to analyse traffic patterns (such as: which lane usually gets more cars at a particular time of the day or which lane usually gets last priority for green light etc).

❖ Simulated WSN

1. *Sensors* – 8 in total (refer to second simulation setup assumption in section 3.2.2.2). Each lane has two sensors each; TQ_T and WT_T
2. *Random Sensor Data Generator Program* – designed a program that would randomly generate sensor data (TQ_T and WT_T) for each lane. (Refer to second simulation assumption in section 3.2.2.2). To ensure

randomness, we employed *ThreadLocalRandom*() which is a sub class of Java's *random()* function.

- *ThreadLocalRandom()* allowed the software system to randomly generate values between a given range for each variable (TQ and WT) respectively (refer to fourth simulation setup assumption in section 3.2.2.2).
- *ThreadLocalRandom*() was preferred over just Java function *random()* and/or *math.random()* because it allows for multiple tasks to use random numbers in parallel and each instance has its own thread. This prevents contention (which can be defined as, conflict over access to a shared resource such as – memory/storage) and improves performance. However, in *random()* function the same instance of *random()* is shared by multiple threads. This leads to contention between multiple threads and so to performance degradation (Kumar, 2012).
- The simulated WSN in this thesis has to perform multiple data assignment to each sensor in each lane concurrently. Hence use of ThreadLocalRandom was appropriate. (See Figure 0-1 in Appendix for code snippet of the random sensor data generator program)

3. *Persistent Storage* – developed a mini program that stores the aggregated sensor data, TQ_T and WT_T in an SQL database. This can later be used by experts in the event they need to analyse traffic patterns (such as: which lane usually gets more cars at a particular time of the day or which lane usually gets last priority for green light etc). Figure 3.4 shows the interface of the application that allows for retrieval and analysis of the stored data.

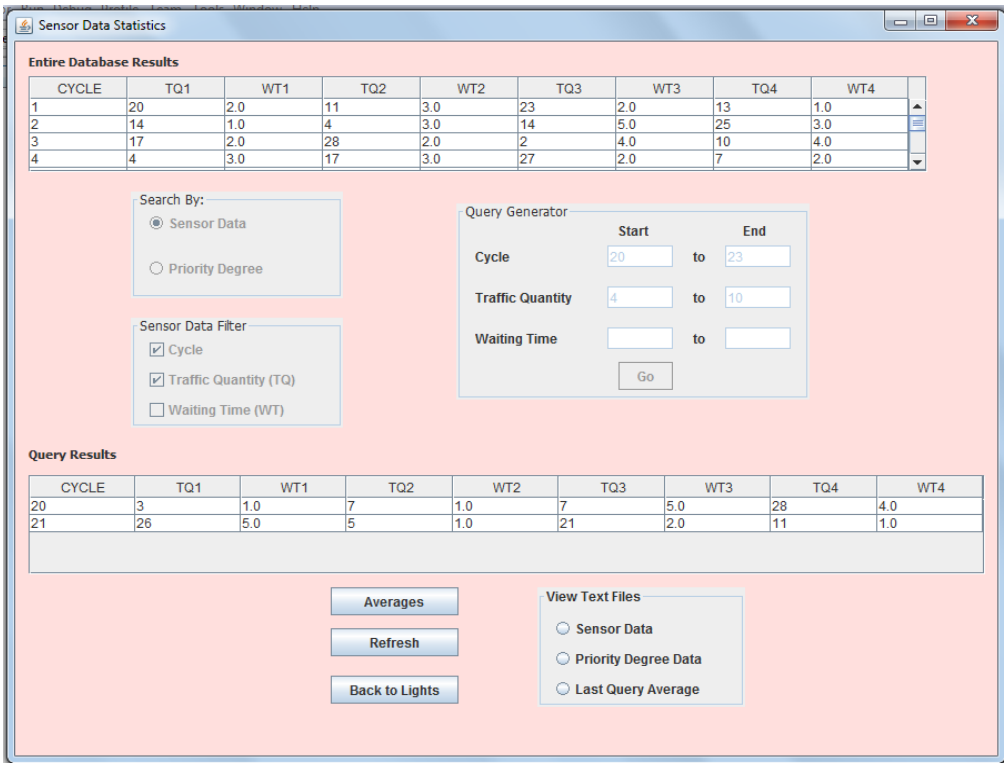


Figure 0.4: Interface for our Persistent Storage for data retrieval/reference

b) Routing Algorithm

We integrated an algorithm in the java application which used the FLC output (PD) to dynamically control traffic flow on all lanes starting from the one with the highest PD. The algorithm is able to make smart decisions such as: if route one higher PD than route two at a particular time then assign green light duration to route one at that moment; if route two has a higher PD than route one then assign green light to route two at that moment. Green light allocation is spontaneous and controlled based on PD value. Figure 3.5 is a pseudo-code that describes a high level view of the proposed algorithm:

```
START
While (model = in run mode) Loop
INPUT PD1, PD2, PD3, PD4
IF (PD1>PD2 & PD1>PD3 & PD1>PD4) THEN
Lane 1 = green
        IF (Lane 1 green_time = 0) THEN
         Lane 1 = red
        IF (PD2>PD3 & PD2>PD4) THEN
         Lane 2 = green
        IF (Lane 2 green_time = 0) THEN
         Lane 2 = red
        IF (PD3>PD4) THEN
         Lane 3 = green
        IF (Lane 3 green_time = 0) THEN
         Lane 3 = red, Lane 4 = green
        IF (Lane 4 green_time = 0) THEN
         Lane 4 = red
        END IF END IF END IF END IF END IF END IF
ELSE IF (PD2>PD1 & PD2>PD3 & PD2>PD4) THEN
Lane 2 = green
        IF (Lane 2 green_time = 0) THEN
        Lane 2 = red
        IF (PD1>PD3 & PD1>PD4) THEN
        Lane 1 = green
        IF (Lane 1 green_time = 0) THEN
        Lane 1 = red
        IF (PD3>PD4) THEN
        Lane 3 = green
        IF (Lane 3 green_time = 0) THEN
        Lane 3 = red, Lane 4 = green
        IF (Lane 4 green_time = 0) THEN
        Lane 4 = red
        END IF END IF END IF END IF END IF END IF
ELSE IF (PD3>PD1 & PD3>PD2 & PD3>PD4) THEN
Lane 3 = green
        IF (Lane 3 green_time = 0) THEN
        Lane 3 = red
        IF (PD1>PD2 & PD1>PD4) THEN
        Lane 1 = green
        IF (Lane 1 green_time = 0) THEN
        Lane 1 = red
        IF (PD2>PD4) THEN
        Lane 2 = green
        IF (Lane 2 green_time = 0) THEN
        Lane 2 = red, Lane 4 = green
        IF (Lane 4 green_time = 0) THEN
        Lane 4 = red
        END IF END IF END IF END IF END IF END IF
ELSE IF (PD4>PD1 & PD4>PD2 & PD4>PD3) THEN
Lane 4 = green
        IF (Lane 4 green_time = 0) THEN
        Lane 4 = red
        IF (PD1>PD2 & PD1>PD3) THEN
        Lane 1 = green
        IF (Lane 1 green_time = 0) THEN
        Lane 1 = red
        IF (PD2>PD3) THEN
        Lane 2 = green
        IF (Lane 2 green_time = 0) THEN
        Lane 2 = red, Lane 3 = green
        IF (Lane 3 green_time = 0) THEN
        Lane 3 = red
        END IF END IF END IF END IF END IF END IF
OUTPUT red signal, amber signal, green signal
End While
STOP
```

Figure 0.5: Smart Traffic Light Routing Algorithm

c) System's Interface Design

In order to demonstrate the approach, it was necessary to simulate the model in one application which was Java. Each approach indicates two sensor input values the TQ_T and WT_T (refer to second simulation setup assumptions in section 3.2.2.2); has a timer (located at the top of the traffic lights) that will count down green light duration of the lane and a traffic light with green, amber and red signals. It also has two command buttons; 'Controller' – which activates the WSN and the FLC functions in an iterative process, 'Analysis' – provides the reference log of traffic data stored.

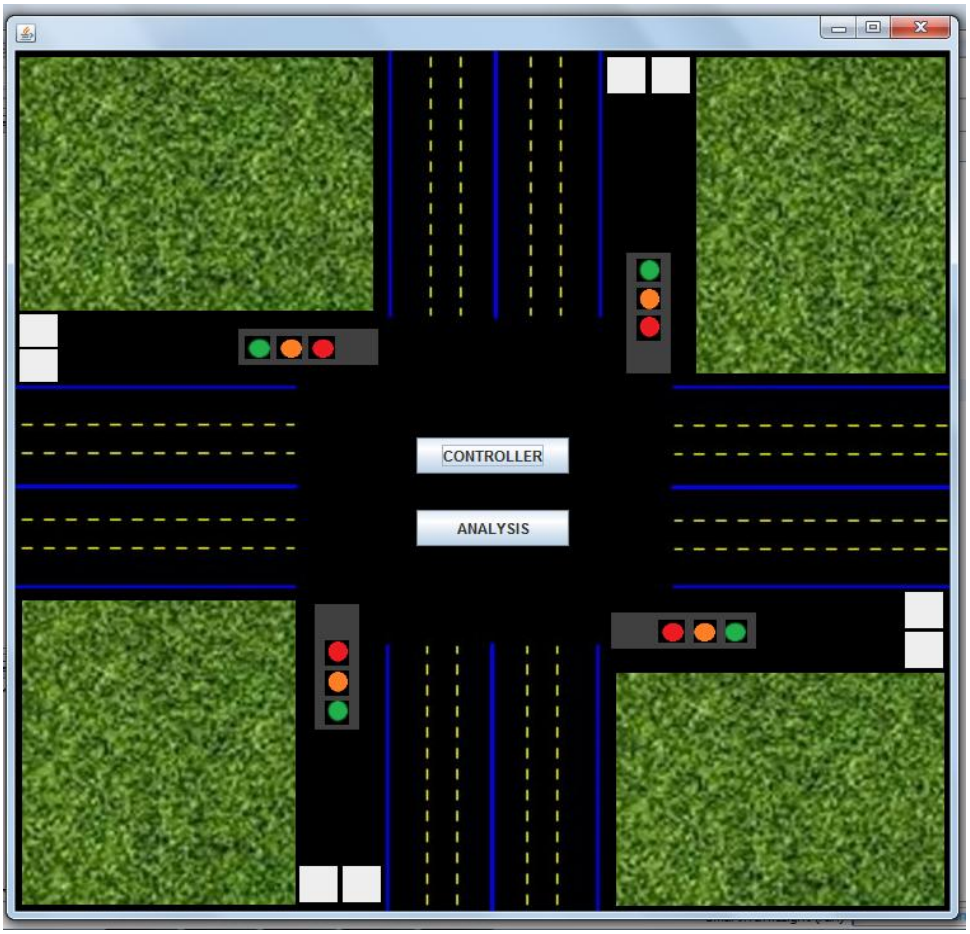


Figure 0.6: Screenshot of the system's interface in Java

ii) qtFuzzylite

This was the software used to design the FLC 'Traffic Light Controller' that responsible for processing the data from the base station (WSN); so as to determine

the order of green light allocation for each approach on the junction. It is an application based on Fuzzylite, a fuzzy logic controller library in C++ developed by Rada-Vilela (2012). This tool was chosen because it is a graphical user interface FLC application, which makes it easier to use and design own FLC than other fuzzy tools such as Matlab (Rada-Vilela, 2011).

For this research, three system variables were designed for the FLC, namely: TQ, WT (inputs) and PD (output). Figure 3.8 – Figure 3.10 describe the graphs we used to design the inputs and outputs in our qtFuzzylite application. Figure 3.11 is a screenshot of how the Traffic Light Controller in qtFuzzylite application looked.

a) Membership Function Distribution

The type of membership function (MF) used to distribute our system variables (TQ, WT and PD) in our simulated FLC is the *Triangular MF* (TriMF).

❖ *Triangular Distribution MF*

It is specified by three parameters, a (minimum limit); b (peak location); and c (maximum limit) which is set to 1. The figure 3.7 describes it:

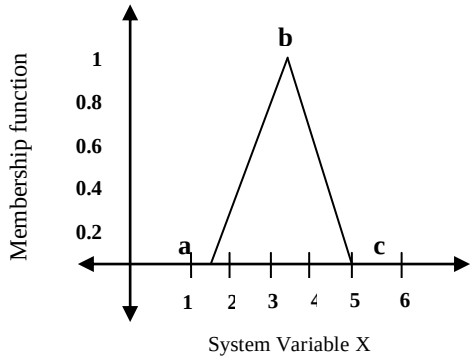


Figure 0.7: Graphical representation of TriMF

The following are the reasons that justify our choice of this type of MF distribution:

- The research had limited access to actual sample data (due to high cost of collection), therefore we used synthetic data. Use of synthetic data allowed us to easily set *a* and *c* limits even though data distribution between these two points was not known. TriMF is also known as '*lack of knowledge'* distribution because of scarcity of data (Hogan, 2015).

- Nature of traffic is uncertain and keeps fluctuating. Meaning it is close to impossible to set a single fixed value for it. Therefore use of TriMF was appropriate as it allowed us to specify the most likely values (estimates) of low(a), high(b) and mode(c). This is reasonable considering the uncertain nature of traffic.

b) System Variables

Inputs and Outputs: The design of an FLC consists of modelling the system inputs and outputs as linguistic variables. A linguistic variable is a set of terms expressed in natural language that can represent the possible values that a system variable can take (Rada-Vilela, 2012).

Our system input variables include: $\boldsymbol{TQ_T}$ and $\boldsymbol{WT_T}$ (refer to equation ***1*** and ***2***) while the output system variable is Priority Degree (***PD).*** The associated terms (linguistic variables) for the three variables are Low (**L**), Medium (**M**) and High (**H**).

These are linguistic variables with ***crisp sets*** because for every possible value, the terms are associated with a degree of either 0 or 1. That is, either the term is associated (1) or it is not (0). For instance, $\boldsymbol{TQ_T}$ can be low and medium (1) but not high (0); or medium and high (1) but not low (0).

Once the inputs are ranked (low, medium or high), they will undergo three stages namely Fuzzification, Inference and Defuzzification and finally the output is obtained. Using the *Triangular Membership Function*, the three parameters (a, b, c) are represented as shown in equation ***3***:

$$\textbf{triangle } (x; a, b, c) = \begin{cases} 0, & x \le a. \\ \frac{x-a}{b-a}, & a \le x \le b. \\ \frac{c-x}{c-b}, & b \le x \le c \\ 0, & c \le x. \end{cases} \qquad \ldots (3)$$

Then to determine the x coordinates of the three corners (a, b, c) of each linguistic term, we modify equation ***3*** using our *max* and *min* limits. Therefore the final formula used to distribute our system variables and their linguistic terms is represented in equation ***4:***

$$triangle\ (x; a, b, c) = max\left[min\left[\frac{x - a}{b - a}, \frac{c - x}{c - b} \right], 0 \right] \quad \ldots (4)$$

❖ *WT*

- The range set for this input system variable is 0 to 6 minutes (refer to second assumption in section 3.2.2.2)

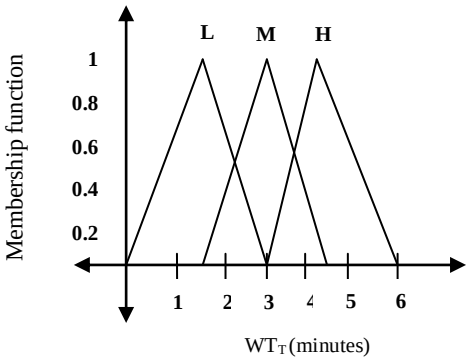


Figure 0.8: Graphical representation of input variable WT_T

❖ *TQ*

- The range set for this input system variable is 0 to 30 cars (refer to second assumption in section 3.2.2.2)

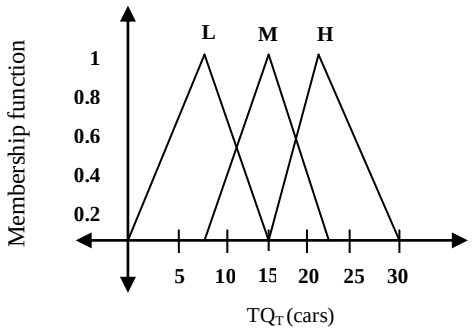


Figure 0.9: Graphical representation of input variable TQ_T

❖ *PD*

- The range set for this output system variable is 0 to 10.

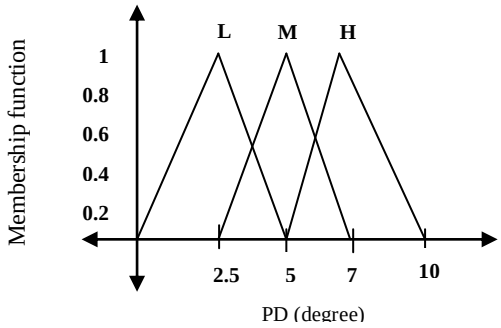


Figure 0.10: Graphical representation of output variable PD

c) FLC interface Design

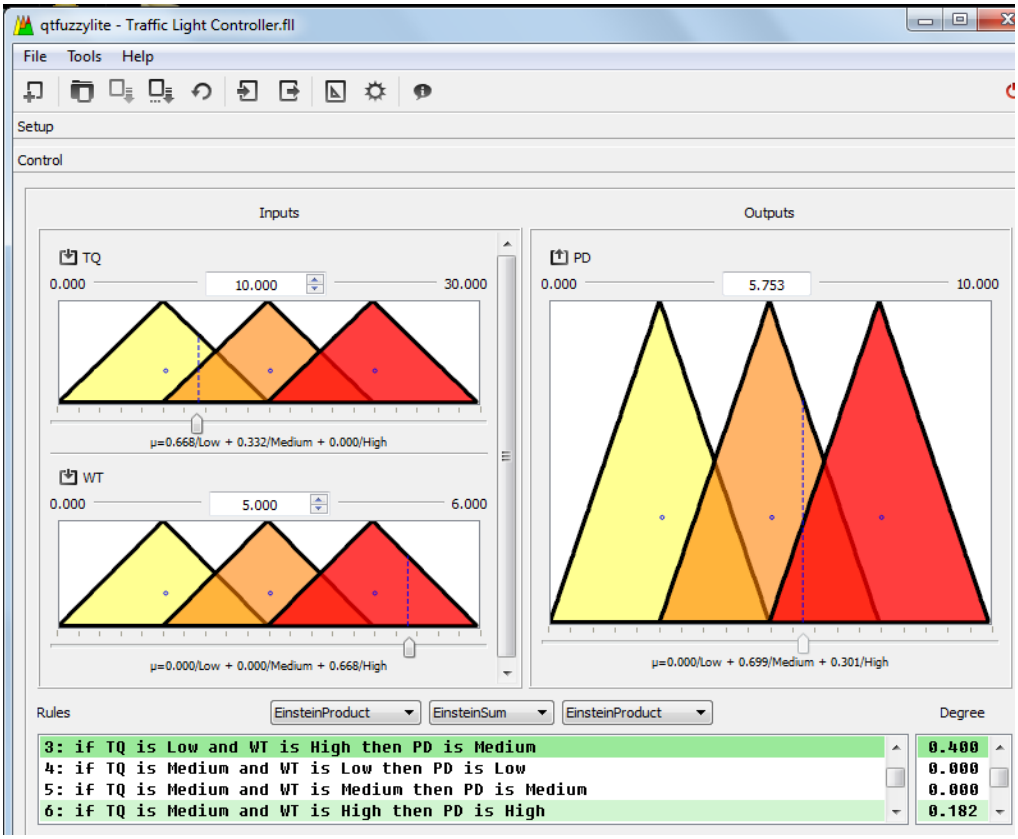


Figure 0.11: Graphical representation of our qtFuzzylite application

The **FLC** developed is responsible for processing the data from the BS so as to determine the order of green light allocation for each approach. This is achieved by using the two inputs, traffic quantity ($\boldsymbol{TQ_T}$) and waiting time ($\mathbf{WT_T}$) to produce the output, priority degree (***PD***). The routing algorithm in Figure 3.5 will then assign green light starting from the approach with the highest PD value. Figure 3.12 describes the stages the inputs of our FLC system undergo after being ranked (low medium high) till an output is derived.

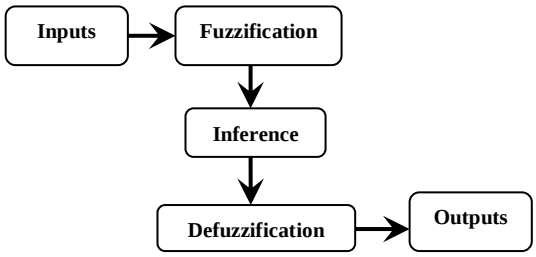


Figure 0.12: FLC Process

1) *Fuzzification*: this is the conversion of crisp input values to ***fuzzy set.*** This provides more information about the system variables by using membership functions (the degree to which a variable is associated with a term) that range between 0 and 1. For instance some variables are ranked with more than one term.

Consider the example in Figure 3.13. It illustrates how crisp set value WT (3.5) that is medium and high but not low is converted to fuzzy set that gives additional degree values of the variable (WT). The associated terms are then described as 0.4 high and 0.7 Medium. These inputs are mapped into fuzzy numbers (i.e. 0.4 and 0.7 respectively) by drawing a line up from the inputs to the input membership functions and marking the intersection point (Princeton, 2007). This is represented as follows:

$$\boldsymbol{WT_T\ (3.5) = 0.0/L + 0.7/M + 0.4/H \ldots (5)}$$

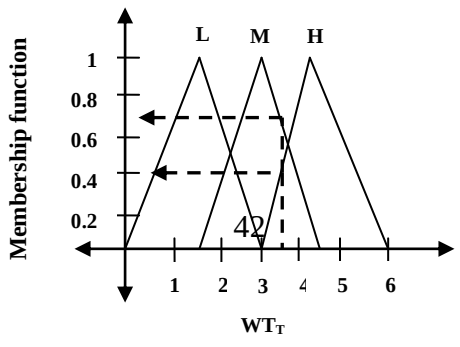


Figure 0.13: Graphical representation of fuzzification

2) *Inference*: at this stage rules are activated that will generate outputs. Inference rules are conditional statements that control the system. Each rule consists of an antecedent and a consequent, each of which comprises of propositions in the form "variable is term". The structure of the antecedent is "***If*** variable is term" while that of the consequent is "***then*** variable is term". When there is more than one proposition in the antecedent, then the propositions can be connected using either the conjunction *and* or disjunction *or*. The antecedent will look as follows: "if variable is term ***and/or*** variable is term" (Rada-Vilela, 2012).

The two commonly used types of inference systems include: Mamdani and Takagi-Sugeno (Rada-Vilela, 2012). These two inference systems are similar during fuzzification, however, they differ when it comes to the way their outputs are determined.

❖ *Mamdani Inference System*

- This inference type expects the *output membership functions* have to be *fuzzy sets*, meaning there is a fuzzy set for each output variable that needs defuzzification (MathWorks); using either centroid and maxima defuzzifiers. Centroid gets the *centre mass* of the accumulation of all fuzzy set output distributions while Maxima gets the *mean* of the accumulation of all fuzzy set output distributions (Princeton, 2007).
- Typical rule in Mamdani fuzzy model has the form:

If (input1 is membership function1) and/or (input2 is membership function2) then (output$_N$ is membership function$_N$)

- The final output in a Mamdani system is represented in the equation ***6***:

$$\text{Final output} = \sum_{i=1}^{n}(x_i \mu_{Ai}(x)) \,/\, \sum_{i=1}^{n}(\mu_{Ai}(x)) \ldots (6)$$

- Where: $x_i \mu_{Ai}(x)$ is the membership function () of the region Ai of the output variable x at a given value/position i^{th} of the output variable x on the x co-ordinate; and n represents the number of rules.

❖ *Takagi-Sugeno*

- *Output membership functions* are either *linear or constants* (MathWorks, 2016). The *output is a crisp number* computed by multiplying each output by a constant and then adding up the results (Princeton, 2007).
- Typical rule in a Sugeno fuzzy model has the form:
  If Input 1 = x and/or Input 2 = y, then Output is z = ax + by + c ... (7)
  where *a*, *b* and *c* in equation ***7*** are user-defined constants.
- Each rule weights its output level, z_i, by the firing strength of the rule, w_i. Using equation ***8*** as example, w_i is computed as follows:

$$w_i = AndMethod(F_1(x), F_2(y)) \ldots \textbf{(8)}$$

where $F_{1,2}(.)$ are the membership functions for Inputs 1 and 2.

- The final output of the system is the weighted average of all rule outputs, computed as (MathWorks, 2016):

$$\text{Final output} = \sum_{i=1}^{n}(w_i z_i) / \sum_{i=1}^{n}(w_i) \ldots \textbf{(9)}$$

Our proposed system uses the Mamdani system. This is because the Sugeno model does not provide a good intuitive method for determining the coefficients (a, b, c in equation ***7***) (Princeton, 2007)). Our Mamdani inference system consists of four logic operators; The T-norm (defines connective and), the S-norm (defines connective or), the activation operator (which is a T-norm) and accumulation operator (which is an S-norm) (Rada-Vilela, 2012; 2014).

Since our system has two inputs ($\boldsymbol{TQ_T}$ and $\boldsymbol{WT_T}$), our generated IF...THEN rules are based on these two inputs and the propositions are connected using the conjunction

and (T-norm operator). Each of these two inputs has *three* linguistic terms (Low, Medium and High). Therefore the number of rules in our fuzzy engine totals to 9 as shown in Figure 3.14. TQ and WT represent the antecedents with their respective membership functions and the PD represents the consequent with its respective membership function.

Rule 1: if TQ is **low** and WT is **low** then PD is **low**
Rule 2: if TQ is **low** and WT is **medium** then PD is **low**
Rule 3: if TQ is **low** and WT is **high** then PD is **medium**
Rule 4: if TQ is **medium** and WT **is low** then PD is **low**
Rule 5: if TQ is **medium** and WT is **medium** then PD is **medium**
Rule 6: if TQ is **medium** and WT is **high** then PD is **high**
Rule 7: if TQ is **high** and WT is **low** then PD is **medium**
Rule 8: if TQ is **high** and WT is **medium** then PD is **high**
Rule 9: if TQ is **high** and WT is **high** then PD is **high**

Figure 0.14: FLC rules for Traffic Light Controller

These rules are generated in qtFuzzylite using the *Operation::increment()* method. It finds all possible combinations of rules based on our linguistic variables. Given a vector $x = <0, 0, 0>$ an upper limits $u = <3, 3, 3>$, the algorithm basically increments the vector x by the unit. The index in the vector refers to the proposition index in the antecedent, and the value refers to the index of the term in the linguistic variable. See Figure 0-5 in the Appendix (Rada-Vilela, 2015).

Fuzzy logic rules with two inputs can also be represented in matrix form to represent *AND* conditions. Table 3.1 describes our 3 × 3 matrix (9 rules) that uses the two inputs, *TQ* and *WT*, and our one output *PD*. One advantage of this matrix representation is that it makes it easy to represent all the rules in a system.

Table 0.1: Matrix representation of the fuzzy set rules

WT / TQ	L	M	H	
L	**L**	**L**	**M**	PD
M	**L**	**M**	**H**	
H	**M**	**H**	**H**	

The fuzzy inputs from the fuzzification stage are taken through 3 steps in the inference stage, namely:

- *Activation of antecedent* – this is where the membership function values derived from the fuzzification stage are assigned to each of their corresponding proposition in the antecedent. These values are then referred to as activation degrees (weight of rule) of the antecedents. Considering our previous example in equation 3 an input value of 3.5 for WT variable, the following Fig. 3.15 describes how the activation degree is assigned:

WT_T (3.5) = 0.0/L + 0.7/M + 0.4/H

Activation

If (***WT_T*** is **low**) = **0.0**
If (***WT_T*** is **medium**) = **0.7**
If (***WT_T*** is **high**)) = **0.4**

Figure 0.15: Activating the antecedent

- *Modification of consequent* – this is done by multiplying the activation degree computed before with the linguistic terms in each of the consequents using an activation operator (which is a T-norm). The Mamdani T-norms that can be used to modify the consequents are the *minimum* and *algebraic product* between two values (Rada-Vilela, 2012). For example if we use our example WT (3.5) fuzzy set, then Figure 3.16 describes in graph form how each consequent is modified

using *minimum* as the activation operator (represented by the ⊗symbol seen in Figure 3.16). The consequents are clipped at the rule strength respectively.

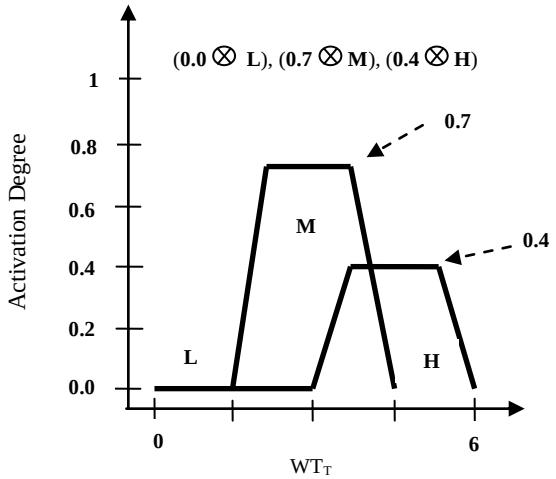


Figure 0.16: Graphical representation of modification of consequents

- *Accumulation of consequents* – this is done by summing up all consequents using an accumulation operator (which is an S-norm). The Mamdani S-norms that can be used here are the *maximum* and *algebraic sum* between two values (Rada-Vilela, 2012). For instance using our example in Figure 3.16; Figure 3.17 describes how the consequents are summed up using the *maximum* as the accumulation operator (represented by the ⊕symbol seen in fig. 3.8).

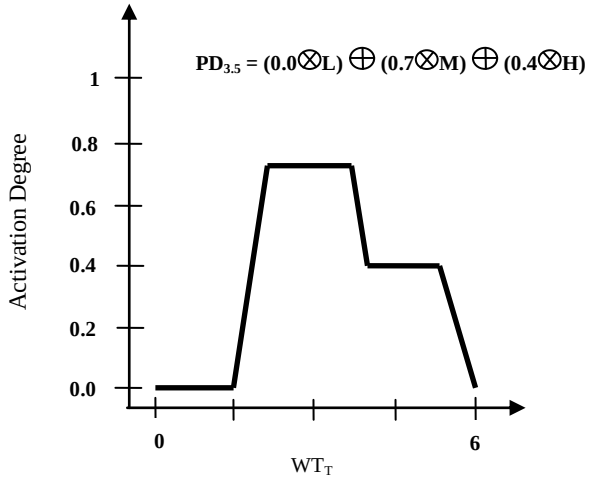


Figure 0.17: Graphical representation of accumulation of consequents

3) *Defuzzification*: this is the process of converting fuzzy outputs above into crisp values (the system output), which are computed using a defuzzifier. Mamdani

uses *centroid* and *maxima* defuzzifiers. For our system we used the centroid which computes the x (WT_T) value of the centre of mass of the fuzzy set to give us the ***PD*** output (Rada-Vilela, 2012). Using Figure 3.17 as an example, the crisp value after centroid defuzzification is the x value indicated in Figure 3.18.

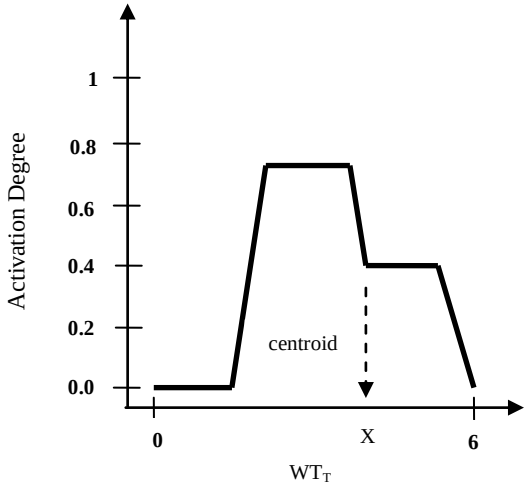


Figure 0.18: Graphical representation of centroid defuzzifier

3.2.2.4 Implementation of the Software System

The following procedure describes how we merge Java and qtFuzzylite into one application:

- Import *jFuzzylite 1.0* library into our java application. This allows us to use the Fuzzylite FLC components within java.
- With the library in place, the next step is to convert the qtFuzzylite code that created our engine from C++ to Java. This feature is provided by the qtFuzzylite application (Rada-Vilela, 2014). We then proceed to export the code to our system in java.
- The random sensor data generator program is then used to randomly generate sensor data (TQ_T and WT_T) for each lane.
- All the sensor data inputs aggregated are then fed into the fuzzy engine which then computes the PD (output) for each lane.
- The routing algorithm then compares the PD value (output generated by the fuzzy engine) of each lane. This algorithm in turn is used to control the traffic

lights and allocates green light starting from the lane with the highest PD to the one with the least.

- To get the green light duration of each lane, we use the product value of TQ_T and WT_T generated for a lane and assume that value to be in seconds. For instance if the TQ for lane 1 is 23 cars and the WT is 3 minutes, then the timer for lane 1 will count down from 69 seconds and not minutes.

$$\boldsymbol{Timer = TQ_T * WT_T \ldots (10)}$$

In Figure 3.19 we see the model in operation. The PD output (on the left side of fig. 3.15) for each lane based on the inputs is as follows:

PD of Lane 1 (TQ_T8, WT_T5.0) = 5.1151727013327415

PD of Lane 2 (TQ_T27, WT_T2.0) = 5.7351853252174845

PD of Lane 3 (TQ_T13, WT_T5.0) = 6.929879693519826

PD of Lane 4 (TQ_T21, WT_T2.0) = 5.3653033450474785

According to the PD values, Lane 3 will be the first to get green light (as indicated), then Lane 2 (indicated with the amber signal), then Lane 4 and finally Lane 1. The green light duration is as described in equation ***10***.

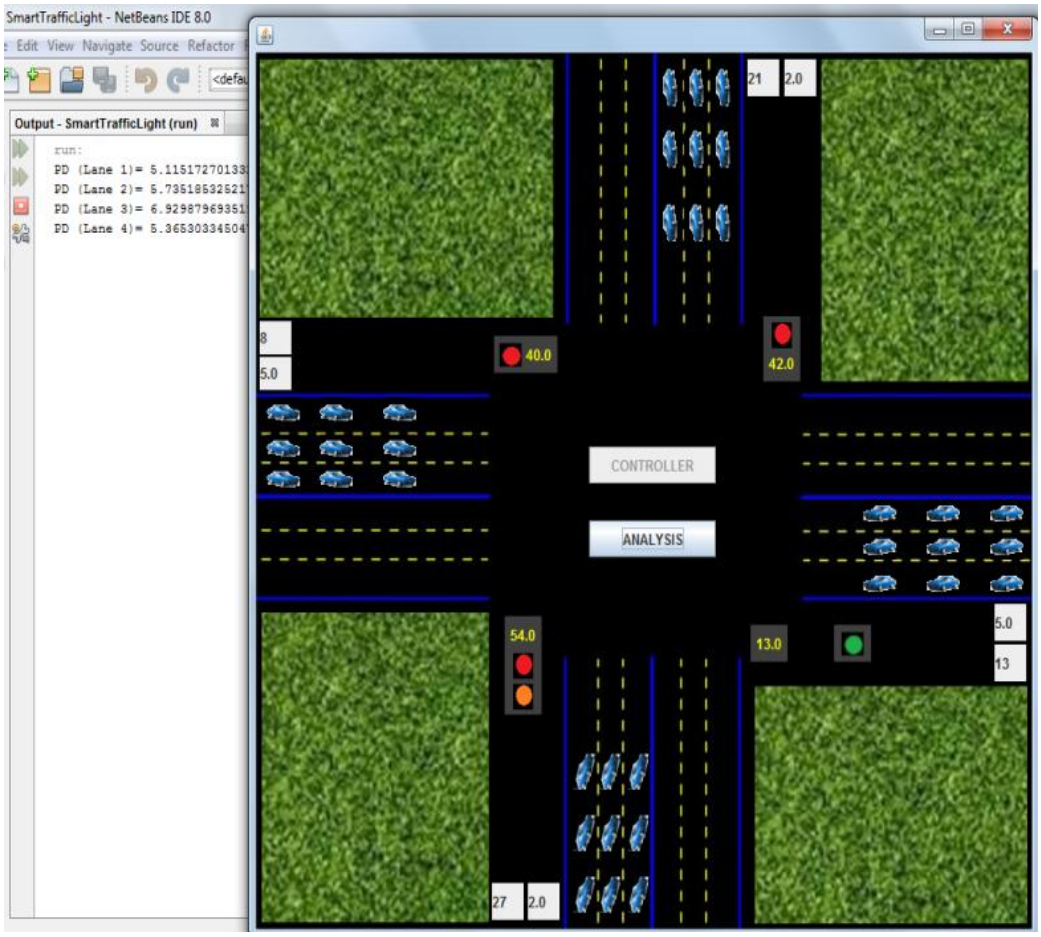


Figure 0.19: Screenshot of our model in operation

Careful consideration was taken during simulation. This is because performing a simulation poses a few risks. For instance, any incorrect key stroke has the potential to alter the results of the simulation and give you the wrong results. Also the fact that the environment is a representation of the real world means that it might not produce 100% correct results always, rather a close estimate. To ensure that the simulation provided accurate results we first ran a base line to prove that it works. We then carried out several similar experiments and checked whether the experimental data sets compare. This was so that we can ascertain that the simulated environment and data derived had credibility.

3.3 Conclusion

This chapter gave a detailed account of the research design and methods/techniques used to collect as well as analyse the data about the research. It also provided an

overview of how the research was carried out and how the data collected was used to develop the proposed approach. It also discussed the tools used to design the model. The research methodology took on the applied research design. This is because we used the data collected from the different tools (observations and literature reviews) to come up with the solution for the traffic jam problem in the urban towns of Nairobi city. To test the hypothesis, we further used a prototype model for the development process. With this prototype model together with a smart routing algorithm, we were able to recreate a virtual environment which provided a test bed to support our hypothesis that, traffic levels can be reduced by developing a smart traffic light controller using fuzzy logic and WSN. The simulated WSN was responsible for collecting data. The FLC was responsible for aggregating the sensor data (inference process) to derive an output (PD) which was then used by the formulated algorithm to intelligently route traffic starting with the lane with the highest PD value. The prototype allowed us to evaluate the performance and effectiveness of the proposed approach before actual adoption.

CHAPTER FOUR
RESEARCH FINDINGS AND DISCUSSION

4.1 Experimental Setup

To evaluate the performance of the proposed approach the model was tested for 100 minutes in total. This was further divided into 5 independent rounds (R1, R2, R3, R4, R5) each consisting of 20 minutes each. The minimum (min) and maximum (max) green light extension; that we allocated for each lane in the simulated junction is 1 second and 3 minutes respectively. The rationale behind this being that the proposed model's architecture assumes that the minimum and maximum number of cars a lane can get is 1 and 30. The following equation ***11*** illustrates how the green extension values were arrived at:

Min/Max. Green extension (seconds) = No. of Cars * min/max. WT… (11)

30 cars * 6 minutes = 180 (seconds) – maximum green time

1 car * 1 minute = 1 (second) – ***minimum green time***

All errors encountered during execution were counted and their details recorded. After every round (20 minutes), the number of cycles and the averages of PD, TQ and WT for each lane and round was calculated. The five rounds were then compared to each other. Finally the overall averages of PD, WT and TQ (accumulation of all rounds) was calculated. Analysis of these overall averages made it possible to evaluate and derive conclusions about the performance of the proposed approach.

To demonstrate this proposed approach we used synthetic data generated by a mini java program as opposed to actual data. Nevertheless, we were still able to measure proposed model's performance and evaluate/simulate how it would operate in real world scenarios. Synthetic data mimic the result set in a far more controlled and accelerated fashion (GENTRIG, 2013). Okeyo *et al*. (2012) prove that using synthetic data can produce system accuracy levels of 83% in activity recognition systems. To test their model, they use a simulation tool to mimic activation of

sensors in a dense sensor based deployment, the simulation tool then plays back the synthetic data generated and feeds this data to the activity recognition system as if the sensor activations are occurring in real-time (Okeyo *et al*., 2012). In this research, synthetic data offered flexibility to test our approach in different scenarios. For instance, we were able to control the volume of the two variables WT and TQ so as to test the model's responsiveness (PD) under different scenarios such as peak (time of the day when the system variables are at their highest) and off-peak (time of day when the system variables are at their lowest). Moreover, using synthetic data we were able to manipulate the data so as to correct any performance errors that came up during testing. Manipulation would not be possible if we used actual data, because the moment its manipulated then it ceases to be actual data. By having control over our experimental data, we were able to accelerate the testing phase of our approach.

4.1.1 Metrics for Evaluation/Performance measures

i) *WT* – evaluated our model's performance by comparing the total WT average derived in the 5 rounds value against maximum (max) and minimum (min) WT value our model can have; which we set to 6 minutes and 1 minute respectively. The model was also evaluated by comparing the effect of WT on PD averages.

ii) *TQ* – we evaluated the model's performance by comparing effect of TQ on PD averages.

iii) *PD* – investigated the model's performance by comparing the total PD average value (derived in the 5 rounds) against the maximum and minimum PD value our model can have; which is 7.500 and 2.500 respectively. This range was derived from the FLC centroid defuzzifier function (see section 3.2.2.2 and fig. 3.9 of this thesis).

iv) *Accuracy and Error margin*– analysed our system performance by setting an 80% benchmark. This means that from the experiments the approach was to record an 80% and above correct decision rate. Equation ***12*** below describes how this figure was calculated. We arrived at this figure by stating that out of the 5 rounds of testing, 4 rounds were meant to run without any errors.

$$^{4}/_{5} * 100\% = 80\% \ldots (12)$$

v) *Fixed time controller* - while we did not design a fixed time controller, we were still able to compare and evaluate the proposed model's performance results to that of a static controller. This is because the timing parameters of a static controller are not dependent on real time data so we were still able to derive simulation results mathematically. The following was the fixed time controller setup for this research:

- We took the proposed model's average WT value 2.985 minutes (see section 4.3 on simulation results) and used that as the fixed green light extension for our static controller. Meaning each lane got 2.985 minutes of green signal per cycle. The red light duration for each lane is therefore 8.955 minutes.
- The red light duration (WT) was derived on the following basis: in a fixed controller, the green light is assigned sequentially from lane 1 to lane 4, so for instance lane 2 will have to wait for lane 3, 4 &1 to complete their green time phase before it (lane2) gets green signal. That means if 2.985 minutes is the green extension for each lane then it means wait time will be number of lanes (N_L) before lane 2 multiplied by green extension (G_T) as shown in equation ***13*** below:.

$$\mathbf{WT = N_L * G_T \ldots (13)}$$

4.2 Observations

Each round had varying number of cycles as well as PD, TQ and WT averages. Table 4-1 to table 4-5 provides the tabulated raw data (TQ and WT values) recorded for each round in all four lanes. R1 had a total of 19 cycles, R2 had 17 cycles, R3 had 12 cycles, R4 had 13 cycles and R5 had 15 cycles.

Table 0.1: R1 data collected from the four approaches for TQ and WT

CYCLE	TQ1	WT1	TQ2	WT2	TQ3	WT3	TQ4	WT4
1	20	2	11	3	23	2	13	1
2	14	1	4	3	14	5	25	3
3	17	2	28	2	2	4	10	4
4	4	3	17	3	27	2	7	2
5	21	4	22	4	22	5	28	2
6	18	3	7	5	13	1	5	1
7	8	3	4	5	4	1	11	3
8	10	4	22	2	1	4	2	2
9	8	5	16	3	3	4	6	1
10	5	4	25	5	29	5	5	2
11	29	4	29	3	16	4	27	1
12	12	5	4	2	18	2	3	2
13	24	4	2	1	25	1	28	1
14	28	4	14	1	11	4	28	1
15	26	3	17	4	18	2	12	3
16	6	4	18	4	11	4	12	3
17	11	1	21	2	14	2	9	4
18	8	3	26	3	25	3	22	4
19	26	5	8	2	19	1	4	5

Table 0.2: R2 data collected from the four approaches for TQ and WT

CYCLE	TQ1	WT1	TQ2	WT2	TQ3	WT3	TQ4	WT4
20	3	1	7	1	7	5	28	4
21	26	5	5	1	21	2	11	1
22	25	3	23	3	25	4	22	1
23	20	2	21	1	15	2	2	3
24	18	2	7	4	20	4	6	4
25	15	1	24	5	11	2	16	3
26	2	2	3	1	26	2	29	1
27	6	2	24	4	18	3	5	5
28	5	1	1	5	24	5	12	5
29	9	1	15	5	29	4	20	3
30	14	4	5	3	22	2	14	2
31	15	5	28	3	26	4	19	2
32	17	5	6	3	29	4	18	2
33	16	3	20	3	13	3	4	1
34	28	1	28	5	8	1	12	3
35	29	4	9	2	28	4	13	2
36	1	5	8	2	21	5	5	1

Table 0.3: R3 data collected from the four approaches for TQ and WT

CYCLE	TQ1	WT1	TQ2	WT2	TQ3	WT3	TQ4	WT4
41	14	1	15	5	15	4	9	2
42	27	1	24	2	11	3	23	1
43	24	1	2	2	8	3	27	2
44	10	5	25	5	16	2	28	4
45	18	2	18	5	7	2	25	2
46	2	4	18	5	22	4	13	2
47	17	4	1	2	3	4	12	3
48	5	3	17	5	19	5	13	1
49	2	5	15	5	18	2	17	4
50	28	2	15	4	19	5	29	2
51	20	4	26	1	24	5	21	5
52	18	1	23	1	12	3	13	1

Table 0.4: R4 data collected from the four approaches for TQ and WT

CYCLE	TQ1	WT1	TQ2	WT2	TQ3	WT3	TQ4	WT4
55	16	3	4	5	6	5	10	2
56	1	5	16	1	13	2	27	2
57	15	4	27	2	2	2	29	5
58	27	5	9	5	1	2	15	4
59	12	1	23	3	3	2	4	4
60	3	1	16	1	17	1	14	5
61	19	5	18	3	12	2	25	5
62	16	2	20	1	24	4	6	1
63	21	3	7	5	26	3	21	2
64	26	3	8	2	13	3	24	5
65	18	2	20	5	21	4	7	1
66	24	5	11	2	2	2	26	1
67	1	1	28	3	28	4	13	5

Table 0.5: R5 data collected from the four approaches for TQ and WT

CYCLE	TQ1	WT1	TQ2	WT2	TQ3	WT3	TQ4	WT4
68	24	5	2	2	11	1	29	4
69	10	5	19	3	19	4	17	4
70	16	5	21	2	27	4	7	3
71	2	4	28	2	19	1	1	1
72	21	4	29	2	4	5	3	2
73	10	4	28	4	19	4	28	1
74	2	2	26	5	20	2	4	2
75	4	4	23	2	27	1	21	3
76	2	2	9	4	4	2	8	2
77	12	2	10	4	4	5	14	5
78	24	4	3	1	14	4	11	2
79	22	5	23	2	3	4	26	3
80	16	5	24	3	28	2	23	3
81	2	3	28	4	3	1	16	4
82	22	3	4	1	12	2	19	5

The tabulated data in table 4.6 to table 4.10 illustrate the computed PD values for each approach in each round.

Table 0.6: R1 computed PD values from the four approaches

CYCLE	PD1	PD2	PD3	PD4
1	5	3.65518583932926	5.7760232696877	2.5
2	2.5	2.5	7.25086204556154	7.5
3	3.81027330880429	5.72751528752005	4.27248471247995	5
4	2.5	5.6049526925302	5.73518532521748	2.5
5	7.37886569690645	7.46559926426004	7.5	5.72751528752005
6	5.96691575364614	5	2.5	2.5
7	2.63213298824023	5	2.5	3.65518583932926
8	5	5.66283598522054	4.27941833988103	2.5
9	5.11517270133274	5.27789637301994	4.26481467478252	2.5
10	4.24676442922586	7.5	7.5	2.5
11	7.5	7.5	6.79504393005762	5
12	6.55268008773708	2.5	4.18287751776121	2.5
13	7.5	2.5	5	5
14	7.5	2.5	5.40704588929525	5
15	7.5	6.89681143976272	4.18287751776121	4.03308424635387
16	4.23606489782067	7.01708090304464	5.40704588929525	4.03308424635387
17	2.5	5.36530334504748	3.20495606994238	4.63469665495252
18	2.63213298824023	7.5	7.5	7.46559926426004
19	7.5	2.53440073573997	3.85168106705396	5

Table 0.7: R2 computed PD values from the four approaches

CYCLE	PD1	PD2	PD3	PD4
20	2.5	2.5	5	7.5
21	7.5	2.5	5.36530334504748	2.5
22	7.5	7.5	7.5	4.88482729866726
23	5	4.59871753734809	3.28266778014845	2.5
24	4.18287751776121	4.2239767303123	7.26997516548039	4.23606489782067
25	2.5	7.5	2.8543427990126	5.27789637301994
26	2.5	2.5	5.74371073015592	5
27	2.5	7.5	5.96691575364614	5
28	2.5	5	7.5	6.55268008773708
29	2.5	7.5	7.5	6.71733221985155
30	6.49449232106269	2.5	5.66283598522054	3.20495606994238
31	7.5	7.50000000000001	7.5	4.59295411070475
32	7.5	2.5	7.5	4.18287751776121
33	5.27789637301994	6.71733221985155	4.3950473074698	2.5
34	5	7.5	2.5	4.03308424635387
35	7.5	2.62113430309355	7.5	3.10318856023728
36	5	2.53440073573997	7.5	2.5

Table 0.8: R3 computed PD values from the four approaches

CYCLE	PD1	PD2	PD3	PD4
41	2.5	7.5	6.71733221985155	2.62113430309355
42	5	5.76393510217934	3.65518583932926	5
43	5	2.5	2.63213298824023	5.73518532521748
44	5.75323557077414	7.5	3.50550767893732	7.5
45	4.18287751776121	7.5	2.5	5.75323557077414
46	4.27248471247995	7.5	7.46559926426004	3.10318856023728
47	6.89681143976272	2.5	4.26481467478252	4.03308424635387
48	2.5	7.5	7.5	2.5
49	5	7.5	4.18287751776121	6.89681143976272
50	5.72751528752005	6.71733221985155	7.5	5.72058166011897
51	7.26997516548039	5	7.5	7.5
52	3.44731991226291	5	4.03308424635387	2.5

Table 0.9: R4 computed PD values from the four approaches

CYCLE	PD1	PD2	PD3	PD4
55	5.27789637301994	5	5	2.73002483451961
56	5	2.74913795443846	3.10318856023728	5.73518532521748
57	6.71733221985155	5.73518532521748	2.5	7.5
58	7.5	5.40128246265191	2.5	6.71733221985155
59	2.5	7.5	2.5	4.25628926984409
60	2.5	2.74913795443846	3.07012030648018	7.25086204556154
61	7.5	5.96691575364614	2.98291909695536	7.5
62	3.50550767893732	4.24676442922586	7.5	2.5
63	7.0639589521767	5	7.5	5.36530334504748
64	7.5	2.53440073573997	4.3950473074698	7.5
65	4.18287751776121	7.5	7.37886569690645	2.5
66	7.5	2.8543427990126	2.5	5
67	2.5	7.50000000000001	7.5	6.92987969351983

Table 0.10: R5 computed PD values from the four approaches

CYCLE	PD1	PD2	PD3	PD4
68	7.5	2.5	2.5	7.5
69	5.75323557077414	6.34481416067074	7.1456572009874	6.89681143976272
70	7.5	5.36530334504748	7.5	2.5
71	4.27248471247995	5.72751528752005	3.85168106705396	2.5
72	7.37886569690645	5.72058166011897	5	2.5
73	5	7.5	7.1456572009874	5
74	2.5	7.5	5	2.5
75	4.25628926984409	5.7760232696877	5	7.0639589521767
76	2.5	4.63469665495252	2.5	2.53440073573997
77	2.98291909695536	5	5	7.25086204556154
78	7.5	2.5	6.49449232106269	2.8543427990126
79	7.5	5.7760232696877	4.26481467478252	7.5
80	7.5	7.5	5.72751528752005	7.5
81	2.5	7.5	2.5	6.79504393005762
82	7.36786701175977	2.5	2.98291909695536	7.5

Table 4.11 describes a summary of the average results we obtained from the experiments. A graphical representation of the relationship between the three system variables, TQ, WT and PD is shown in fig. 4.1.

Table 0.11: Average results of the data collected on TQ, WT and PD for all four lanes in 76 cycles for the five rounds

TABULATED RESULTS OF THE EXPERIMENTS															
Name	TQR1	WTR1	PDR1	TQR2	WTR2	PDR2	TQR3	WTR3	PDR3	TQR4	WTR4	PDR4	TQR5	WTR5	PDR5
Lane 1	15.526	3.368	5.135	14.647	2.765	4.909	15.417	2.75	4.796	15.308	3.077	5.327	12.6	3.8	5.467
Lane 2	15.526	3	5.143	13.765	3	4.894	16.583	3.5	6.04	15.923	2.923	4.98	18.467	2.733	5.456
Lane 3	15.526	2.947	5.111	20.176	3.294	5.914	14.5	3.5	5.121	12.923	2.769	4.495	14.267	2.8	4.841
Lane 4	13.526	2.368	4.187	13.882	2.529	4.37	19.167	2.417	4.905	17	3.231	5.499	15.133	2.933	5.226
Totals	60.104	11.683	19.576	62.47	11.588	20.087	65.667	12.167	20.862	61.154	12.0	20.301	60.467	12.266	20.99
Average	15.026	2.921	4.894	15.618	2.897	5.022	16.417	3.042	5.216	15.289	3.0	5.075	15.117	3.067	5.248

The Chart in fig. 4.1 provides a summary of the average distribution of WT and PD with relation to TQ across the five rounds of tests.

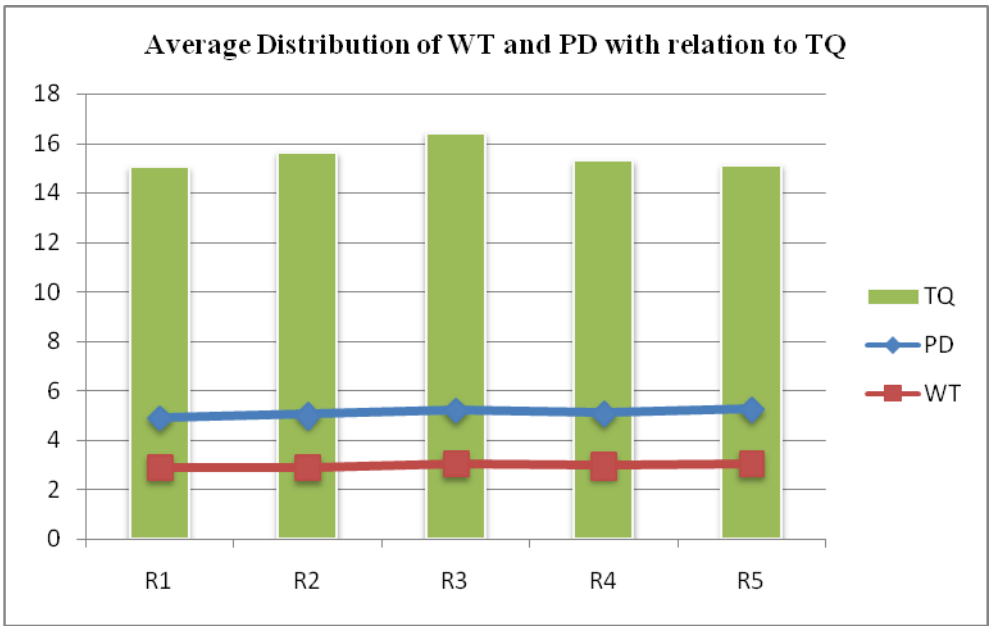


Figure 0.1: Average distribution of WT and PD with relation to TQ across the 5 rounds

The chart in fig. 4.2 describes the average PD distribution for all lanes in each round with a total of 76 cycles (19, 17, 12, 13 and 15 cycles for R1, R2, R3, R4, and R5 respectively). The columns indicate the individual PD average registered for each lane per round. For instance in R1 lane 1 recorded a PD average of 5.135 (see column PD1 in table 4.11). The line graph indicates the PD average registered in each round after accumulating all the individual lane PD averages in a round. For instance in R1 all four lanes recorded an average PD of 4.894.

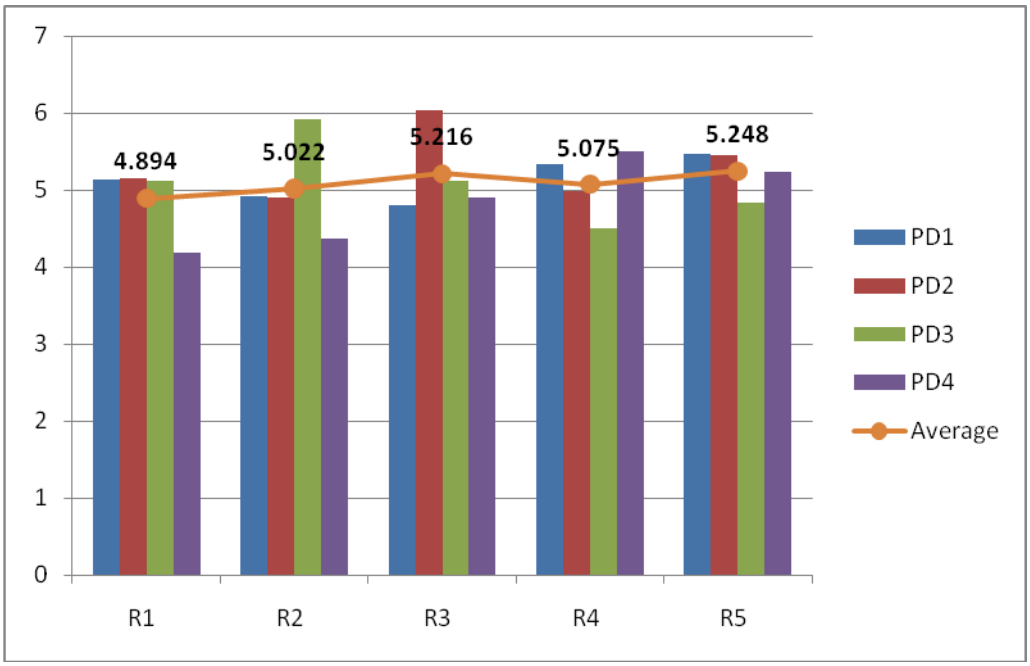


Figure 0.2: Average PD distribution for all lanes in each round

4.3 Simulation Results

From the simulation our system recorded a total of 76 cycles; 19, 17, 12, 13 and 15 cycles for R1, R2, R3, R4 and R5 respectively. There were 4 errors recorded; 2 errors from R1 and 2 from R4 in cycles 6, 10, 61 and 67 respectively. All the errors were as a result of the fuzzy engine inference shooting wrong PD. For instance, in cycle 10, lane 2 was given green light before lane 3 yet lane 2 had TQ=25 and WT=5.0 and lane 3 had TQ=29 and WT= 5.0 (refer to Table 4.1, 4.4, 4.6, 4.9 and 4.12 for more information on raw data of these cycles). However, the error was negligible.

Table 0.12: Description summary of the errors encountered

Item	*R1*	*R2*	*R3*	*R4*	*R5*
No. of errors	2	nil	nil	2	nil
Cycles with errors	6 and 10	nil	nil	61 and 67	nil
Error description	Wrong PD value assignment for lane 3 & 4 and 2 & 3 respectively.	nil	nil	Wrong PD value assignment for lane 1 & 4 and lane 2 & 3 respectively.	nil
Error margin	Negligible - 0.000000000000001 and 0.000000000000004 respectively	nil	nil	Negligible - 0.000000000000001 and 0.000000000000009 respectively	nil

We also observed that the highest WT resulted in highest PD in 4 rounds (R2, R3, R4 and R5) out of the 5 rounds; highest TQ on the other hand resulted in high PD in 3 rounds (R1, R2 and R4). Further, lowest WT resulted in lowest PD in 3 rounds (R1, R2 and R4) while lowest TQ resulted in lowest PD in 2 rounds (R4 and R1). This is described in table 4.13.

Table 0.13: Comparison of highest and lowest TQ and WT averages with relation to highest and lowest PD averages for all 5 rounds

	HIGHEST		**LOWEST**	
	TQ	**WT**	**TQ**	**WT**
PD R1	✓	**X**	✓	✓
PD R2	✓	✓	**X**	✓
PD R3	**X**	✓	**X**	**X**
PD R4	✓	✓	✓	✓
PD R5	**X**	✓	**X**	**X**

The model registered an average PD of 5.091; this was an average PD accumulation of all rounds (see table 4.11 average row); which was at the mid mark of our PD metric range (2.500 to 7.500). A WT average of 2.985 minutes was recorded, meaning on average the lanes usually wait 2.985 minutes before getting green light which was at the mid way mark of our WT metric range (1 minute to 6 minutes); for each lane in our simulated junction.

Lastly our system recorded a 94.7% correct decision rate. This was calculated by deducting the error margin which was obtained as shown in equation ***14***:

$$Error\ magin = \frac{No.of\ Errors}{Total\ No.\ of\ Cycles} * 100\% \ldots (14)$$

$$^{4}/_{76} * 100 = 5.3\%$$

4.4 Discussion

From the experiments, the following theories were derived:

i) Wait Time (WT) influences nature of traffic (i.e. traffic flow and traffic congestion) more than traffic quantity (TQ). For instance, when it comes to traffic congestion, a high WT means possible congestion in the lanes. The longer the cars wait for green light the more TQ increases because cars from other intersections join the lane. For this reason, there is a need to increase traffic flow. This is supported by the fact that 4 out of 5 rounds a high WT resulted in a high priority degree (PD). Meaning on average anytime the lanes registered a high WT they got green light priority. A high TQ on the other hand registered a high PD only 3 out of 5 rounds. Further, we can also conclude that WT also influences traffic flow because; a high PD influenced by high WT will trigger green light signal (movement of cars) in the lanes that record the high PD; also a low WT means increased traffic flow because cars wait less time to get movement signal (green). This is supported by the fact that in 3 out of 5 rounds a low WT resulted in a low PD compared to 2 out of 5 that a low TQ resulted in low PD.

ii) A low TQ does not mean there is no congestion on the lanes. This is supported by the fact that a low TQ influenced a low PD 2 out of the 5 rounds.

iii) Starvation can be resolved by ensuring that TQ is not ignored no matter how low it is and WT is not prolonged. For as long as a lane has cars regardless of the quantity it should not be skipped. From our experiments we are able to prove that our approach caters for starvation because 3 out of 5 rounds a low TQ did not register the lowest PD value. In R2 a low TQ was awarded 3rd priority, in R3 it was awarded 2nd priority and in R5 it was awarded 1st priority. Also our approach ensured that WT was not prolonged because in 4 out 5 rounds; anytime a lane registered a high WT it was awarded high PD (1st green light priority).

iv) If a traffic light is to effectively manage traffic lights at a 4 way intersection, it needs to incorporate both WT and TQ in deciding red and green light allocation. This is supported by the fact that, as much as WT came top factor in determining green light allocation, we observed that TQ was not far behind. The margins were separated by just one round. For instance: 3 out f 5 rounds a high TQ resulted in a high PD while a high WT resulted in high PD 4 out of 5. Also a low TQ resulted in a low PD 2 out of 5 rounds while a low WT resulted in low PD in 3 out of 5 rounds.

v) Smart traffic lights perform better than the fixed time lights. This is supported by the fact that our system recorded an average WT of 2.985 minutes while that of the fixed time controller was 8.955 minutes. As mentioned earlier a low WT results in increased traffic flow because the cars do not wait too long to get movement signal (green). Therefore, our approach is better at controlling and managing traffic flow and consequently congestion within the junction.

4.5 Conclusion

From our simulations, the benefits are two-fold; firstly, by registering an average WT of 2.985 minutes which was below the 6 minutes maximum WT limit; it means that the proposed approach successfully managed to increase traffic flow/movement of

cars uniformly for all lanes. This is evident in fig. 4.1 which indicates a fairly balanced WT curve across all approaches in the five rounds. A low WT value results in increase in traffic flow since cars do not wait too long for green light. The WT also being within the stated maximum green extension time of 3 minutes meant that our systems' functionality was operationally sound; if the average WT was above the green extension then it would mean that the proposed model allows lanes to wait longer than the stipulated green extension which is not ideal and will consequently increase congestion. Second benefit is that our approach managed to address the issue of starvation. Our PD value of 5.091 being right at the middle of the PD range expected (2.500 -7.500) meant that there was fair allocation of green light among the lanes (see fig. 4.1 and fig. 4.2). The PD average has a fairly balanced curve across all lanes and rounds. In conclusion, from the experiments and comparison to a fixed time controller we are able to support the fact that a smart traffic light controller can autonomously and efficiently control and manage traffic at a four-way intersection.

CHAPTER FIVE
CONCLUSION AND RECOMMENDATIONS FOR FUTURE WORK

5.1 Conclusion

This research was set out to find a sustainable solution to the traffic congestion problem faced in most urban cities. Our research mainly targeted Kenya, particularly its capital city Nairobi; where the traffic congestion is so inflated that it was ranked 5th place in the world (IBM, 2012) among the countries with the worst traffic jams. One solution Kenya has adopted since launch of Kenya's Vision 2030 in 2006 is the construction of new roads (GoK, 2007). However, this approach has not proven to be so effective in managing traffic in Nairobi for a number of reasons. First reason being that the rate at which travellers buy cars has surpassed that of new infrastructure, this is consistent with Downs' law (1962) that increasing road supply invariably increases vehicle traffic. Second reason, construction of roads is quite costly, most developing countries such as Nairobi are not able to sustain this solution without incurring debts. Thirdly due to effects of urbanization, getting the space to construct these roads on has become an issue. One area gaining interest in ATMS is the use of artificial intelligent systems/techniques to make traffic routing decisions. These systems include; FES, ANN and WSN. These systems use real time data and try to mimic human reasoning thus prove promising in vehicle traffic control and management. In this research we presented a novel model of a smart traffic light control using fuzzy logic and wireless sensor networks to resolve or at least minimize this traffic congestion problem.

We successfully managed to achieve all the research objectives. The first and second objectives were realized through studying of related works. The parameters considered during related work selection was on traffic data collection methods used (such as sensor types), communication methods applied (such as multi-hop or single-hop) to transmit data and data aggregation techniques (such as offline versus real-time, adaptive versus non-adaptive or hybrid strategies). The third, fourth and fifth

objectives were achieved through the design of a simulation model of the FLC, WSN and junction using Java; this made it possible to demonstrate and evaluate effectiveness of the proposed approach.

The outcome of this research based on our experiments was three-fold. Firstly we successfully managed to demonstrate that a smart traffic light controller performs better than a static controller when it comes to reducing average waiting times at intersections. The dynamic nature of the proposed approach, i.e. ability to assign green light based on prevailing traffic conditions, meant it was better tuned to control the ever changing and uncertain traffic conditions on the roads. Secondly, the experiment results showed that the smart traffic light control model can fairly and uniformly route and manage traffic flow at a four-way intersection. This is evidenced by the fact that on average all the lanes registered a WT and PD values that were close to or at the mid way mark of their respective ranges. This meant that the waiting time and green light allocation was fairly distributed among the 4 lanes of the junction hence resolving issue of starvation. Lastly the approach used in this research was able to demonstrate smart traffic routing decisions (assigning green light based on urgency/priority), consequently minimizing the need of having traffic personnel on the roads.

This research not only proposed a solution to traffic congestion but also contributed to the existing body of knowledge by introducing a novel approach for smart traffic control. The experience we gained during the research was a valuable asset to our on-going interest in the fields of pervasive computing and artificial intelligence. It provided an excellent platform to learn how smart systems can be embedded in day to day objects allowing them to communicate and co-operate in accomplishing tasks.

5.2 Recommendation for future work

Currently our proposed approach does not include the synchronization of traffic lights among the different junctions, such that, roundabout 1 operates independently of roundabout 2 decisions. A future recommendation for the research is to implement a model that controls traffic in more than one junction. This can be achieved by using

already available infrastructure (such as the CCTVs already installed in most of the Nairobi city highways). Also incorporate emergency cases (such as, ambulances, road accidents) as input variables in determining green light allocation and duration. In addition, the data collected (logged) should be analysed to derive other input variables in determining green light; such as day of the week and time of day (peak/off peak).

REFERENCES

Barbosa, M. R., & Pinto, G. C. (2010). Exploring the use of Neural Networks in urban traffic management. *Business Sustainability I*, 287-292. Retrieved from http://labve.dps.uminho.pt/bs11/CD/PDF/47%20-%20pp%20287-292%20-%20Pinto%20G.C.,%20Barbosa%20M.R..pdf

Beattie, A. (2011). What is the difference between artificial intelligence and neural networks?. *techopedia*. Retrieved from http://www.techopedia.com/2/27888/programming/what-is-the-difference-between-artificial-intelligence-and-neural-networks

Bhuvaneswari, P.T.V., Arun raj, G.V., Balaji, R., Kanagasabai, S. (2012). Adaptive Traffic Signal Flow Control using Wireless Sensor Networks. *IEEE Computer Society*, 85-89. Paper presented at 4th International Conference on Computer Intelligence and Communication Networks. Retrieved from http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6375077&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6375077

CBT. (2012). *Going backwards the new road programme*. Retrieved from http://bettertransport.org.uk/sites/default/files/research-files/Roads_to_Nowhere_October2012_web_spreads_0.pdf

Dai, Y., Hu, J., Zhao, D., & Zhu, F. (2011). Neural network based online traffic signal controller design with reinforcement training. *Intelligent Transportation Systems (ITSC)*, 1045-1050. Paper presented at 14th International IEEE Conference, Washington, DC. Retrieved from http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6083027&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6083027

Dargie, W. & Poellabauer, C. (2010). Fundamentals of Wireless Sensor Networks Theory And Practice (1st ed.). John Wiley and Sons Ltd.

Duranton, G., & Turner, M. A. (2011). The Fundamental Law of Road Congestion: Evidence from US cities. University of Toronto Department of Economics. Retrieved from http://www.economics.utoronto.ca/public/workingPapers/tecipa-370.pdf

Downs, A. 1962. The law of peak hour expressway congestion. *Traffic Quarterly 16*(3) 393–409. Retrieved from http://worldcat.org/issn/00410713

Fahmy, M. M. M. (2007). An Adaptive Traffic Signalling for Roundabout with Four-Approach Intersections Based on Fuzzy Logic. *Journal of Computing and Information Technology (CIT), 15* (1), 33-45. Retrieved from http://cit.srce.unizg.hr/index.php/CIT/article/view/1625/1329

GENTRIG (2013). Synthetic test data...reality is overrated. *Generating Right Test objects.* Retrieved from www.gentrigo.com

GoK. (2007). *Kenya Vision 2030, The Popular Version.* Retrieved from http://www.vision2030.go.ke/cms/vds/Popular_Version.pdf

Hawi, R., Okeyo, G. & Kimwele, M. (2015). *International Journal of Computer Applications Technology and Research (IJCAT), 4*(7), 566-573. Retrieved from http://www.ijcat.com/archives/volume4/issue7/ijcatr04071014.pdf

Hogan, R. (2015). "Probability Distributions for Measurement Uncertainty". Retrieve from http://www.isobudgets.com/probability-distributions-for-measurement-uncertainty/

IBM. (2012). *A vision of a smarter city*. Retrieved from http://www-05.ibm.com/za/office/pdf/IBM_-_A_Vision_of_a_Smarter_City_-_Nairobi.pdf

IBM. (2011). *Frustration Rising: IBM 2011 Commuter Pain Survey*. Retrieved from http://www-03.ibm.com/press/us/en/attachment/35314.wss?fileId=ATTACH_FILE1&fileName=Frustration%20Rising_IBM%202011%20Commuter%20Pain%20Report.pdf

IBM. (2013). IBM Opens Doors of First African Research lab – Continent's Grand Challenges in its Sights. *IBM News room*. Retrieved from http://www-03.ibm.com/press/us/en/pressrelease/42409.wss

KIPPRA. (2013). *Kenya Economic Report.* Retrieved from http://www.kippra.org/downloads/Kenya%20Economic%20Report%202013.pdf

Khiang, T. K., Khalid, M., & Yusof, R. (1997). Intelligent Traffic Lights Control by Fuzzy Logic. *Centre for Artificial Intelligence and Robotics*. Retrieved from http://www.researchgate.net/profile/rubiyah_yusof/publication/229029935_intelligent_traffic_lights_control_by_fuzzy_logic/file/9c960527c4a1870fae.pdf

Kumar, M. (2012). Musings, Thoughts and Cogitations on Java – Prefer ThreadLocalRandom over Random. Retrieved from

http://thoughtfuljava.blogspot.co.ke/2012/09/prefer-threadlocalrandom-over-random.html

Knaian, A. (2000). A wireless sensor network for smart roadbeds and intelligent transportation systems. *Master's Thesis, Massachusetts Institute of Technology*. Retrieved from http://resenv.media.mit.edu/pubs/theses/AraKnaian-Thesis.pdf

Kumar, R. (2011). Vehicle-Actuated Signal Controller For Heterogeneous Traffic Having Limited Lane Discipline *Institute of Transportation Engineers (ITE) Journal*. Retrieved from http://www.researchgate.net/publication/259533291

Lecue, F., Tucker, R., Bicer, V., Tommasi, P., Tallevi-Diotallevi, S. & Sbodio, M. (n.d). Predicting Severity of Road Traffic Congestion using Semantic Web Technologies. *IBM Research, Smarter Cities Technology Centre, Damastown Industrial Estate, Dublin, Ireland*. Retrieved from http://2014.eswc-conferences.org/sites/default/files/papers/paper_205.pdf

MathWorks (2016). Fuzzy Inference Systems. Retrieved from http://www.mathworks.com/help/fuzzy/mamdani-fuzzy-inference-systems.html

Mittal, P. K., & Singh, Y. P. (2013). Analysis and Designing of Proposed Intelligent Road Traffic Congestion Control System with Image Mosaicking Technique. *International Journal of IT, Engineering and Applied Sciences Research (IJIEASR), 2* (4), 27-31. Retrieved from http://www.irjcjournals.org/ijieasr/apr2013/7.pdf

Munassar, N. M. & Govardhan, A. (2010). A Comparison between Five Models of Software Engineering. *International Journal of Computer Science Issues (IJCSI), 7* (5), 94-101. Retrieved from http://www.ijcsi.org/papers/7-5-94-101.pdf

Ndonga, S. (2014). Gridlock in Nairobi as traffic lights put to test. Retrieved from http://www.capitalfm.co.ke/news/2014/02/gridlock-in-nairobi-as-traffic-lights-put-to-test/

Okeyo, G., Chen, L., Wang, H. & Sterritt, R. (2012). Dynamic sensor data segmentation for real-time knowledge-driven activity recognitio. *Pervasive and Mobile Computing (PMC), 10*(B), 155-172. Retrieved from http://www.sciencedirect.com/science/article/pii/S1574119212001393

Pascale, A., Nicoli, M., Deflorio, F., Dalla Chiara, B., & Spagnolini, U. (2012). Wireless sensor networks for traffic management and road safety. *IET Intelligent Transport Systems, 6*(1), 67-77. Retrieved from

http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6157101&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6157101

Patel, M., & Ranganathan, N. (2001). IDUTC: An Intelligent Decision-Making System for Urban Traffic-Control Applications. *IEEE Transactions on Vehicular Technology, 50*(3), 816-829. Retrieved from http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=933315&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D933315

Princeton, University. (2007). Fuzzy inference systems. Retrieved from http://www.cs.princeton.edu/courses/archive/fall07/cos436/HIDDEN/Knapp/fuzzy004.htm

Rada-Vilela, J (2012). Fuzzylite-paper 3.1 a fuzzy logic control library in C++. Retrieved from http://www.fuzzylite.com.

Rada-Vilela, J (2014) Fuzzylite: a fuzzy logic control library. Retrieved from http://www.fuzzylite.com.

Rada-Vilela, J (2015) Fuzzylite fl Operation.h. Retrieved from http://github.com/fuzzylite/fuzzylite/blob/master/fuzzylite/fl/Operation.h

Runyoro, A., Zlotnikova, I. & Ko, J. (2014). Towards Automated Road Information Framework a Case Study of Tanzania. *Transport and Telecommunication, 15*(1), 12-19. Retrieved from http://www.tsi.lv/sites/default/files/editor/science/Research_journals/Tr_Tel/2014/volume15_issue_1_02_paper.pdf

Srinivasan, D., Choy, M. C., & Cheu, R. L. (2006). Neural Networks for Real-time Traffic Signal Control. *IEEE Transactions on Intelligent Transportation Systems, 7* (3), 261-272. Retrieved from http://www.jhuapl.edu/spsa/PDF-SPSA/Srinivasan_etal_IEEETITS06.pdf

W3C. (2013). Semantic web – Ontologies. Retrieved from http://www.w3.org/standards/semanticweb/ontology

Wen, W. (2008). A dynamic and automatic traffic light control expert system for solving the road congestion problem. *Expert Systems with Applications, 34(4*), 2370-2381. Retrieved from https://www.academia.edu/3253331/A_dynamic_and_automatic_traffic_light_control_expert_system_for_solving_the_road_congestion_problem

Yokota, T. (2004). ITSS Technical Note 1 For Developing Countries. *ITS Technical Notes.* Retrieved from http://www.worldbank.org/transport/roads/its%20docs/ITS%20Note%201.pdf

Yousef, K. M., Al-Karaki, J. M., & Shatnawi, A. M. (2010). Intelligent Traffic Light Flow Control System Using Wireless Sensor Networks. *Journal of Information Science and Enginering, 26,* 753-768. Retrieved from http://www.iis.sinica.edu.tw/page/jise/2010/201005_02.pdf

Zadeh, L.A. (1965). Fuzzy sets. *Information and Control, 8(3), 338–353*. Retrieved from http://www.cs.berkeley.edu/~zadeh/papers/Fuzzy%20Sets-Information%20and%20Control-1965.pdf

Zhou, J., Chen Philip, C.L., Chen Long. & Zhao, W. (2013). A User-Customizable Urban Traffic Information Collection Method Based on Wireless Sensor Networks. *IEEE Transaction on Intelligent Transportation Systems, 14*(3), 1119-1128. Retrieved from http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6493443

APPENDIX

Important Software System Code Snippets

a) Simulated WSN – Random Sensor Data Generator Program

```
//Randomly generate TQT sensor data values
TQ1 = (ThreadLocalRandom.current().nextInt(1, 29));
TQ2 = (ThreadLocalRandom.current().nextInt(1, 29));
TQ3 = (ThreadLocalRandom.current().nextInt(1, 29));
TQ4 = (ThreadLocalRandom.current().nextInt(1, 29));

//Display the values
lblTQ3.setText(String.valueOf(TQ3));
lblTQ4.setText(String.valueOf(TQ4));
lblTQ2.setText(String.valueOf(TQ2));
lblTQ1.setText(String.valueOf(TQ1));

//Randomly generate WTT sensor data values
WT1 = (ThreadLocalRandom.current().nextInt(1, 5));
WT2 = (ThreadLocalRandom.current().nextInt(1, 5));
WT3 = (ThreadLocalRandom.current().nextInt(1, 5));
WT4 = (ThreadLocalRandom.current().nextInt(1, 5));

//Display the values
lblWT3.setText(String.valueOf(WT3));
lblWT4.setText(String.valueOf(WT4));
lblWT2.setText(String.valueOf(WT2));
lblWT1.setText(String.valueOf(WT1));
```

Figure 0-1: Generation of synthetic sensor data for TQ and WT of all four roundabout lanes using ThreadLocalRandom() method

b) Entire FLC process carried out by the actionPerformed() operation of inner class MyListener.

```
engine.getInputVariable("TQ").setInputValue(TQ1);
engine.getInputVariable("WT").setInputValue(WT1);
engine.process();
output1 = engine.getOutputVariable("PD").defuzzify();
System.out.println("PD (Lane 1)= " + output1);

engine.getInputVariable("TQ").setInputValue(TQ2);
engine.getInputVariable("WT").setInputValue(WT2);
engine.process();
output2 = engine.getOutputVariable("PD").defuzzify();
System.out.println("PD (Lane 2)= " + output2);

engine.getInputVariable("TQ").setInputValue(TQ3);
engine.getInputVariable("WT").setInputValue(WT3);
engine.process();
output3 = engine.getOutputVariable("PD").defuzzify();
System.out.println("PD (Lane 3)= " + output3);

engine.getInputVariable("TQ").setInputValue(TQ4);
engine.getInputVariable("WT").setInputValue(WT4);
engine.process();
output4 = engine.getOutputVariable("PD").defuzzify();
System.out.println("PD (Lane 4)= " + output4);
```

Figure 0-2: FLC operations – setting inputs (TQ and WT) of all four lanes, fuzzification of the inputs, inference process and defuzzification

c) Calculating green light duration

```
public void CalculateTicker() {
  TickResult1 = TQ1 * WT1;
  TickResult2 = TQ2 * WT2;
  TickResult3 = TQ3 * WT3;
  TickResult4 = TQ4 * WT4;

lblTick1.setText(String.valueOf(TickResult1));
lblTick2.setText(String.valueOf(TickResult2));
lblTick3.setText(String.valueOf(TickResult3));
lblTick4.setText(String.valueOf(TickResult4));
}
```

Figure 0-3: Calculating green light duration for all four lanes by multiplying the respective values of their two inputs variables (TQ and WT)

d) Automatic looping of the software system's operations to ensure autonomous system execution

```
public void repeat(){

if(lblTick1.getText().equals("0")&&lblTick2.getText().equals("0")&&
lblTick3.getText().equals("0")&&lblTick4.getText().equals("0")){
            Controller.setEnabled(true);
            Controller.doClick();
        }
 }
```

Figure 0-4: Looping all software system processes to ensure autonomous system re-execution (iteration)

e) 'Traffic Light Controller' (FLC) rule extraction and generator

```
inline bool Operation::increment(std::vector<int>& x,
std::vector<int>&min, std::vector<int>& max){
      return increment(x, -1 + x.size(), min, max);
}

inline bool Operation::increment(std::vector<int>& x, int
position, std::vector<int>& min, std::vector<int>& max){

      if(x.empty() or position<0) return true;
      bool overflow = false;

      if (x.at(position) < max.at(position)) {
        ++x.at(position);
      }else{
            overflow = (position= =0);
            x.at(position) = min.at(position);
            --position;

      if(position >=0){
      overflow = increment(x, postion, min, max);
      }
      }
            return overflow;
}
```

Figure 0-5: Operation::increment() method that generates our 9 fuzzy system's rules