M.G. KANGUET ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI YÜKSEK LİSANS TEZİ
MUĞLA 2019

T.C.

MUĞLA SITKI KOÇMAN UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DESIGN AND IMPLEMENTATION OF AN ADAPTIVE TRAFFIC CONTROL SYSTEM

MASTER OF SCIENCE

MARTINIEN GLOIRE KANGUET

OCTOBER 2019

MUĞLA

T.C.

MUĞLA SITKI KOÇMAN UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DESIGN AND IMPLEMENTATION OF AN ADAPTIVE TRAFFIC CONTROL SYSTEM

MASTER OF SCIENCE

MARTINIEN GLOIRE KANGUET

OCTOBER 2019

MUĞLA

MUGLA SITKI KOÇMAN UNIVERSITY

Graduate School of Natural and Applied Sciences

APPROVAL OF THE THESIS

The thesis submitted by **MARTINIEN GLOIRE KANGUET** with the title of " **DESIGN AND IMPLEMENTATION OF AN ADAPTIVE TRAFFIC CONTROL SYSTEM** " has been unanimously accepted by jury members on the 18th of October, 2019 to fulfil the requirements for the degree of Master of Science in the Department of Electrical and Electronics Engineering.

THESIS JURY MEMBERS

Prof. Dr. Sırrı Sunay Gürleyük **(Head of Jury)** Signature:

Department of Electrical and Electronics Engineering,
Muğla Sıtkı Koçman University, Muğla

Assist. Prof. Dr. Fatma Yıldız Taşcıkaraoğlu **(Thesis Advisor)** Signature:

Department of Electrical and Electronics Engineering,
Muğla Sıtkı Koçman University, Muğla

Assist. Prof. Dr. Göker Aksoy **(Member)** Signature:

Department of Civil Engineering,
Işık University, Istanbul

APPROVAL OF THE HEAD OF THE DEPARTMENT

Assoc. Prof. Dr. Bahadır Süleyman Yıldırım Signature:

Head of Department, Electrical and Electronics Engineering,
Muğla Sıtkı Koçman University, Muğla

Assist. Prof. Dr. Fatma Yıldız Taşcıkaraoğlu Signature:

Thesis Advisor, Department of Electrical and Electronics Engineering,
Muğla Sıtkı Koçman University, Muğla

Date of Defense: 18/10/2019

Hereby, I declare that all the results and information presented in this thesis have been obtained as required by the academic and scientific ethical rules. In addition, I declare that all ideas, conclusions, and information that are not original to this study were referred.

Martinien Gloire Kanguet

18/10/2019

ÖZET

ADAPTİF BİR TRAFİK KONTROL SİSTEMİ TASARIMI VE UYGULAMASI

Martininien Gloire KANGUET

Yüksek Lisans Tezi

Fen Bilimleri Enstitüsü

Elektrik Elektronik Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Fatma Yıldız TAŞCIKARAOĞLU

Ekim 2019, 108 sayfa

Kentsel ulaşım sistemlerinin halkın sosyal ve ekonomik durumunun iyileştirilmesine etkili bir şekilde katkıda bulunması için gerekenin yapılmaması trafik sıkışıklığının önemli sebeplerinden biridir. Bu trafik tıkanıklığı sorununu çözmenin en kolay yolu kavşaklardaki trafik ışıklarını uygun bir şekilde kontrol etmektir.

Günümüzde, dünyadaki çoğu şehir sabit zamanlı trafik kontrol sistemlerini kullanmaktadır. Bu sistemlerde, önceden toplanmış trafik verileri kullanılarak sinyal planları günün farklı zamanlarına göre tasarlanırken, kavşaklardaki trafik taleplerindeki beklenmedik değişiklikler veya trafik akışının zamanla değişmesi tıkanıklığa neden olmaktadır.

Kavşaklarda seyahat sürelerini ve gecikmeleri azaltmayı hedefleyen bu çalışmada, koordineli çalışan üç kavşaklı trafik ağının uyarlanabilir kontrolü için bir algoritma geliştirilmiştir. Arduino kart denetleyicileri tarafından kontrol edilen üç kavşaklı bir trafik prototipi oluşturularak uygulama yapılmıştır. Kavşaklar arasında iletişim sağlamak için seri iletişim teknolojisi uygulanmıştır. Ek olarak, Arduino yazılımından gelen seri monitör komutları, kavşaklardan gelen trafik verilerini görüntülemek üzere programlanmıştır. Prototip üzerinde deneyler yapılarak elde edilen sonuçlar sabit zamanlı trafik kontrol sisteminden alınan test sonuçlarıyla karşılaştırılmıştır. Deneysel sonuçlar, değişen trafik dinamiklerine göre önerilen sistemin etkinliğini ve gerçek bir trafik ağında uygulanabilme imkanını göstermektedir.

Anahtar Kelimeler: Adaptif Trafik kontrolü, Koordineli kavşak trafik kontrolü, Gerçek-zamanlı trafik kontrol uygulaması, Arduino uygulaması

ABSTRACT

DESIGN AND IMPLEMENTATION OF AN ADAPTIVE TRAFFIC CONTROL SYSTEM

Martininien Gloire KANGUET

Master of Science (M.Sc.)

Graduate School of Natural and Applied Sciences

Department of Electrical and Electronics Engineering

Supervisor: Assist. Prof. Dr. Fatma Yıldız TAŞCIKARAOĞLU

October 2019, 108 pages

Traffic congestion is one of the causes of the inefficiency of the urban transport systems to contribute effectively to the people's social and economic situation improvement. The easiest way to solve this congestion problem is to control the intersection traffic lights in an appropriate way.

In most cities worldwide Fixed-time traffic control systems are widely used. In these systems, signal plans are designed according to different times of the day using pre-collected traffic data, while traffic demands or traffic flow at intersections change over time, resulting in congestion.

In this study, which aims to reduce travel times and delays at intersections, an adaptive control algorithm is developed for a coordinated three-intersection traffic network. A three-intersection traffic prototype, controlled by the Arduino board controllers is implemented in real time. For providing communication between the intersections, a serial communication technology is implemented. In addition, serial monitor commands from the Arduino software are programmed to display the traffic data received from the intersections. Experimental tests are performed on the prototype, and the results are benchmarked against test results from a fixed-time traffic control system. The experimental results show the efficiency of the proposed system with dynamic demands and varying flow and ability to be applied to a real-time traffic network.

Keywords: Adaptive traffic control, Coordinated intersection traffic Control, Fixed Control, Implementation of real-time traffic control, Traffic Prototype with Arduino Board

To my dear family

ACKNOWLEDGEMENTS

As the author of this thesis, I would like to express my greatest gratitude to my advisor, Assist. Prof. Dr. Fatma YILDIZ TAŞCIKARAOĞLU, for her advice, encouragement, and guidance. Without her personal involvement and assistance, this study would not have been completed.

I am very grateful to Assoc. Prof. Dr. Pınar DOĞAN, who gave me significant advice and a favorable agreement to get start my Master degree in the department of Electrical and Electronics Engineering.

I am also very grateful to Assist. Prof. Dr. Akın TAŞCIKARAOĞLU, who provided me important guidance for the traffic result comparison methods.

I would like to express my deep gratitude to the TÜBİTAK organization, which supported this thesis project number 117E182.

I would also like to express my deep gratitude to my cousin, Ardeche MAVOUNGOU-MBOUNGOU for his encouragement and support, and to my friend, Faruk ÖZDEMİR, with whom we hardly worked together to construct the propotype proposed in this thesis.

TABLE OF C ONTENTS

LIST OF TABLES

LIST OF FIGURES

SYMBOLS AND ABBREVIATIONS

C_i	Distance from detector to stop line
C_{Opt}	Optimal cycle length
d	Distance from detector to stop line
g_i	Effective green time for Phase i
$G_{\min}$	Minimum green time
G_i	Green time for phase i
G_t	Net green time or split green
L	Total lost time
P	Passage time interval
S_i	Saturation flow rate
vc	Sum of critical lane volumes
vc_i	Critical lane volume for Phase i
vi	Observed traffic flow for phase i
tL	Start-up lost time
V	Sum of critical line volume
Y	Sum of flow ratio
KΩ	Kilos ohm
Vph	Vehicle Per Hour
KB	Kilo Bytes
KB/S	Kilo Bytes per Second
VPH	Vehicle Per Hour
GDP	Gross Domestic Product

MHz	Megahertz
VP5M	Vehicle Per Five Minutes
ITMS	Intelligent Traffic Management System
ATCS	Adaptive Traffic Control System
VCC	Voltage at the Common Collector
PWM	Pulse Width Modulation
AC	Alternating Current
DC	Direct Current
RF	Radio Frequency
IR	Infrared
SRAM	Static Random Access Memory
IoT	Internet of Things
UTS	Urban Transportation System
UTSC	Urban Traffic Signal Control
ITS	Intelligent Traffic Signal
ITSS	Intelligent Traffic Signal System
IDE	Integrated development environment
SPI	Serial Peripheral Interface
Qty	Quantity

1. INTRODUCTION

This chapter provides a general view of the content of our study. Thus, it discusses about the thesis background and problem statement, and describes the thesis objectives, scope, originality and contributions.

1.1. Background

Urban Transport Systems (UTS) effectively contribute to the development of cities. They improve the social and economic situation of people by allowing them to move from one place to another, which facilitates the production of goods and services, as well as their exchanges or trades with other places such as countries, cities, villages, etc. (Davol, A. P. 2001; Al-Mudhaffar, A. 2006). These systems may be more effective and profitable if they have adequate infrastructure and effective traffic control systems. Thus, Ineffective traffic controls, inadequate traffic infrastructure and the growth in vehicle's numbers on the road have been shown to be the main causes of traffic congestion (Al-Mudhaffar, A. 2006).

The number of people living in cities is growing rapidly around the world. A statistical study shows that the number of people living in the cities exceeded 50%, which will increase in 2030 and more in 2050 (Rajsman, M. and Horvat, R. 2014). This growth is leading to an increase in economic activities that contribute positively to the increase in national GDP. Also, it is positively changing the conditions and lifestyles of people and giving them the possibilities to buy personal vehicles. As result, this is leading to a rapid increase in the number of vehicles in the cities, which increases traffic congestion (Anonym, 2012).

The increase of traffic congestion causes the serious social problems evaluated in terms of health, lost time (long travel time), and money. In terms of health, traffic congestion leads to the consumption of fuel and to the noise generated by vehicles, which pollutes environment (Ketabdari, M. 2013). Air is a vital resource for human life and it should be of good quality, unfortunately this atmospheric pollution reduces its quality thus leading to serious consequences for the health of people (Li, M. and Mallat, L. 2018). Added to this, are the stress and frustration caused by the traffic, which affect the health of drivers increasing the rate of traffic accidents (Jiao, P. 2016). In terms of lost time and money, traffic congestion causes the lost of time for ,motorists and passengers leading to delays in arriving at work, in educational institutions, business locations, etc., which negatively impact economic activities (Jiao, P. 2016). So, all of these effects negatively affect productivity and reduce the people's contribution in national GDP (Anonym, 2012).

Most municipal authorities use the fixed-traffic control system, and often build new traffic road networks in order to deal with congestion problems. Unfortunately, this solution is inefficient, expensive, and inadequate since the new roads built become more and more congested due to the continuous increase of vehicles (Anonym, 2012; Jiao, P. 2016).
Therefore, in view of all these realities, one of the most effective solution is to develop adaptive traffic control systems that do not require the specific enhancements of existing traffic infrastructures. Also, which are able to respond dynamically to the traffic demands in order to reduce traffic congestion with its negative effects (e.g. air pollution, long travel time etc.).

1.2. Problem Statement

There are generally two systems used to control the traffic signal at intersections. The first is a fixed-time control system, which is the most used around the world (Jiao, P. 2016). Its timing plan is obtained offline. Also, it has some predefined parameters such as cycle length, phase splits and green time duration making it unable to respond efficiently to the

real-time demands of traffic. This result in the increase of travel times, fuel consumptions, air pollutions and road accidents (Aslania, M., Mesgari, M.S. and Wieringb, M. 2017). The second is an adaptive control system that continuously measures traffic flow or density. This allows it to respond dynamically to traffic demands in real time, resulting in reducing travel times, fuel consumptions, air pollutions and road accidents. Therefore, instead of implementing a predetermined control system, it was better for us to design and implementation an Intelligent Traffic Control (ITC), specifically an Adaptive Traffic Control System (ATCS), which will be dynamically coordinated.

1.3. Study Objective and Scope

1.3.1. Objective

Due to the limitations of the fixed-time traffic control systems, and to negative effects of traffic congestion mentioned previously, this thesis mainly aims to reduce travel times and delays, which will inevitably reduce fuel consumption, air pollution and road accidents at coordinated intersections. Its objectives are the following:

- To develop an adaptive control algorithm for a coordinated traffic network,
- To perform a real-time traffic control,
- To implement a coordinated traffic network prototype with three intersections,
- To establish communication between the intersections, and
- To design a traffic data monitoring system.

1.3.2. Scope

The relationship between our thesis, its objectives, and scope is done since the scope includes:

- Use of critical intersection approach applied within the proposed algorithm in order to computes optimal cycle length and phase green durations. Also, a coordinated

timing plan is applied, which establishes a green wave through the intersections, intends to minimize average delays, and maintains it under saturated conditions.

- Use of three interconnected Arduino board controllers that adjust and calculate the duration of traffic signals based on real-time traffic flow measurements (e.g. traffic vehicle number). In addition, they assign more green signal duration to the line with the highest traffic vehicle number (The most crowded road lines).
- Use and understanding of a serial communication command integrated within the Arduino IDE software that permits the controllers to transmit traffic data such as the vehicle number to each other. Added to this, is the use of a serial monitor, which provides the possibility to display the transmitted traffic data.

1.4. Thesis Originality and Contribution

This thesis contains an original value that we have not found in the old papers, and books studied in the course of our deep literature review periods. This original value is therefore a simple general guideline that covers the design, implementation and experimentation procedures of a traffic control system, that combines the features of two control strategies (adaptive and coordinated) within an algorithm. To this originality is added deep and clear explanations on the control of three connected intersections that exchange information in real time, as well as on the function of displaying traffic data exchanged between the controllers, that can be seen from a remote control center.

Designing and implementation of an adaptive control system applied on isolated intersection is very difficult due to many parameters that it requires. This becomes more difficult when the control system is applied on coordinated intersections forming a traffic network. Thus, the simple guideline proposed by this thesis will contribute to the literature, (i) by providing deep literature reviews related to adaptive control projects, (ii) by assisting engineers and students in designing and implementing of appropriate signal timing plans for intelligent traffic system, as well as (iii) by inspiring them on the devices required in the construction of a porotype of an arterial traffic network.

1.5. Thesis Organization

This thesis includes six chapters organized as follows; chapter 1 discuss about the thesis background and problem statement and describes the study objectives, scope, originality and contributions. Chapter 2 provides literature reviews related to existing fixed-time, adaptive, and coordinated traffic control strategies. It also presents literature reviews of adaptive control projects using the Arduino boards as main controllers. Chapter 3 provides an overview of the Arduino boards. Chapter 4 describes the proposed traffic control, which involves the adaptive algorithm development, as well as the prototype implementation. Chapter 5 describes the test procedures and presents the test results. Chapter 6 discusses the conclusion of this thesis, the recommendation, as well as the ideas related to the future works.

2. LITERATURE REVEW

Literature review of traffic control systems was the first important step performed in this study in order to get a great background of the proposed topic, to reveal importance of this thesis, and to determine the existing gap it should fill. To these ends, the review was focused on deep study of articles, books and theses dealing with the subject related to traffic controls as well as with the traffic control projects based on the Arduino board controllers.

This chapter first presents a brief history of traffic control systems. Then it deeply discusses about the traffic control system basics. Finally, it describes the works related to traffic control strategies and projects. Moreover, it provides a conclusion in which a point of view related to the existing traffic controls is given.

2.1. Brief History of Traffic Control Systems

Traffic control systems are usually placed at the urban intersections to regulate and control the traffic. This control authorizes or forbids drivers and pedestrians to pass through intersections under certain conditions. This rule of control is applicable throughout the world (McShane, C. 1999). The first control system used in most cities around the world was a manual control system. In fact, it consisted of a hand semaphore mounted on a raised support and placed in the middle of the intersection. On this were mounted two panels, one indicated the sign "Go" while the other indicated the sign "Stop". It was generally used in London and New York (Shelby and Gebhart, S. 2001).

In 1868, the English citizen Westminster brought a significant improvement of the semaphore model by developing the first colored traffic lights control. This system

consisted of only two gaslights (green and red), which was first installed and tested in London. Its operations was of very short period, because the gas used in the lights exploded. Thus, this led to the suspension of the use traffic control system for many decades (Al-Mudhaffar, A. 2006; McShane, C. 1999).

In 1908, Henry Ford introduced the first T-control model, whose its production and use increased in 1913. This did not solve the congestion problem due to the increase in the vehicle numbers on the roads (McShane, C. 1999).

In the early 1917s, An American citizen William Potts brought a significant revolution in the history of control systems. Firstly, he added a third yellow light making the three-colored control system (green, yellow and red). New York was the first where this system was tested. Secondly, he implemented the first control system with four directions in 1920. Finally, the use of this system was widely spread in European cities between at the end of the 1921s (McShane, C. 1999; Al-Mudhaffar, A. 2006).

Another very significant evolution of control systems was possible in 1922, thanks to the implementation of automatic timers developed by American electrical engineers. This led to automatically timed traffic control systems that automated the semaphore model by introducing an automatic timer in the systems. Houston was the first city where this system was installed and tested. It was therefore possible to establish control plans with fixed duration and well-defined cycle lengths in which light traffic could be turned "OFF" or "ON" while taking into account certain traffic constraints (Shelby and Gebhart, S. 2001).

Nowadays, this control system is still used. Significant improvements are observed in control policies as well as in the performance of timers, depending on the controller types (McShane, C. 1999; Al-Mudhaffar, A. 2006).

2.2. Traffic Control System Basics

The main role of traffic control system is to ensure the safety and efficiency of the movements of people and goods at traffic intersections, as well as along of traffic networks. This role may be effectively accomplished if the control system is properly designed and implemented. Thus, we can talk about advantages and disadvantages of traffic control systems (Smallwood, G., Schijns, S., Tedesco, M. and McCormick, M.O. 2001).

Traffic control system consist of sets of infrastructure and technological devices. Its brain is a controller in which control policies are programmed. Generally, these policies define the time duration of cycle length, green, yellow and red lights. Also, they define how movement should be allocated to phases, and how the controllers should operate under certain traffic conditions. The most visible devices of traffic control systems are traffic lights, which are generally placed at intersections (Badjie, O., Khalid, O. and Ali, O.M. 2016; McShane, C. 1999).

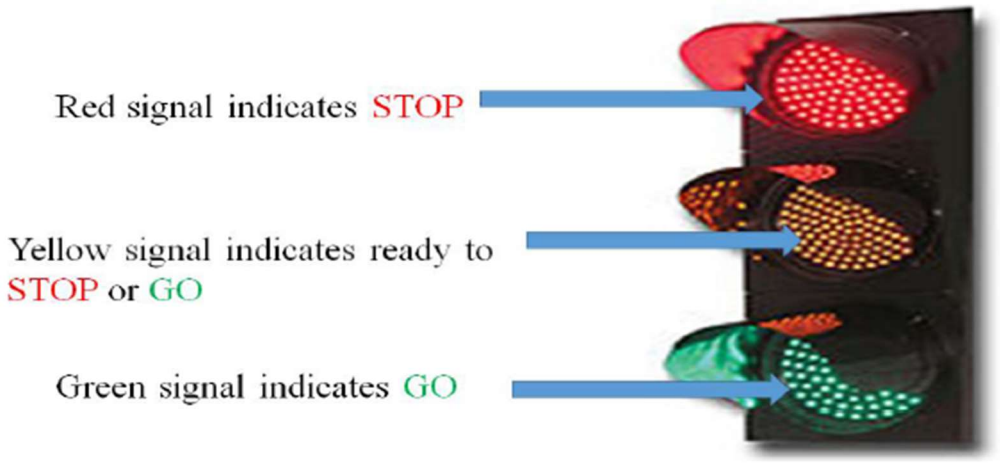

Figure 2.1. Traffic lights (Badjie, O., Khalid, O. and Ali, O.M. 2016)

2.2.1. Advantages and disadvantages of traffic control systems

According to Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. (2011), a well-designed traffic control systems should be based on needs and studies, thus it should contribute to the:

- Reduction of traffic accidents and collisions,
- Attribution of orderly movement of traffic,
- Reduction of travel time, delay , and traffic noise,
- Reduction of air pollution and fuel consumption,
- Increase of intersection capacity provision of signal continuity in the case of coordinated control,
- Increase of traffic safety,
- Contribution to the development of cities, and
- Improvement in economic situation of people.

According to MnDOT (2017), a not well-designed traffic control system should negatively contributes to the:

- Increase of traffic delay and fuel consumption,
- Increase of accidents and collisions,
- Increase of traffic noise,
- Decrease of traffic safety, and
- Increase of motorist's frustration.

2.2.2. Variables and phase movements of traffic control systems

The common fundamental variables of a traffic control system include cycle length, phase, interval, split, and offset. Their definitions are summarized as presented in Table 2.1., and their arrangement is depicted below (see Figure 2.2.).

Table 2.1. Cycle variables definition
(Robert, L., Gordon, P.E., Warren-Tighe, P.E. 2005)

Variable	Definition
Cycle Length	The time required for one complete sequence of signal intervals (phases).
Phase	The portion of a signal cycle allocated to any single combination of one or more traffic movements simultaneously receiving the right-of-way during one or more intervals.
Interval	A discrete portion of the signal cycle during which the signal indications (pedestrian or vehicle) remain unchanged.
Split	The percentage of a cycle length allocated to each of the various phases in a signal cycle.
Offset	The time relationship, expressed in seconds or percent of cycle length, determined by the difference between a defined point in the coordinated green and a system reference point.

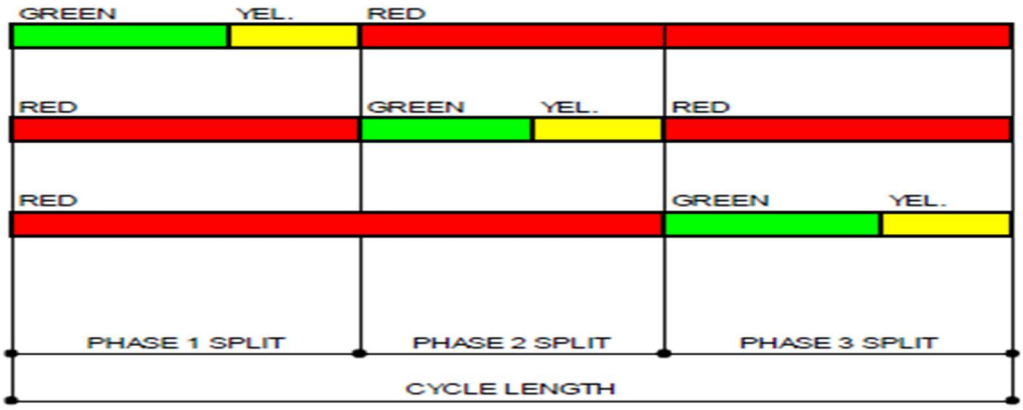

Figure 2.2. Cycle variables arrangement (MnDOT, 2017)

In addition, some physical variables of traffic such as vehicle presence, flow rate, occupancy, density, speed, headway, and queue length are generally detected by detectors, or they are estimated by engineers in order to establish efficient dynamic control plans (Robert, L., Gordon, P.E., Warren-Tighe, P.E. 2005).

As shown below, Figure 2.3. presents two-phase signal movement, and figure 2.4. depicts three-phase signal movement.

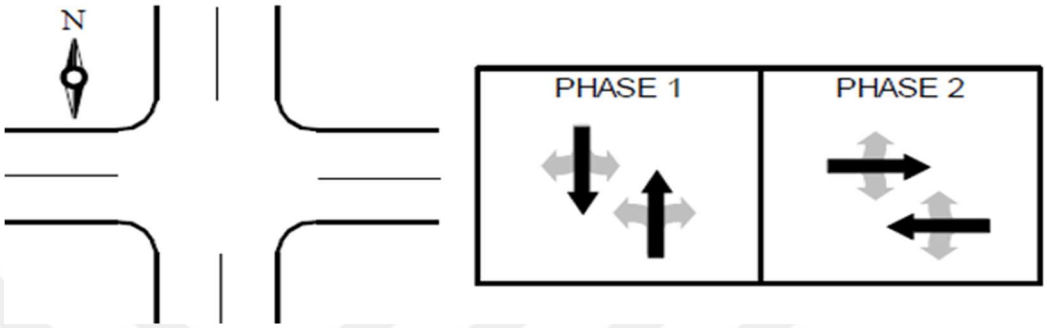

Figure 2.3. Two-phase signal movement (MnDOT, 2017)

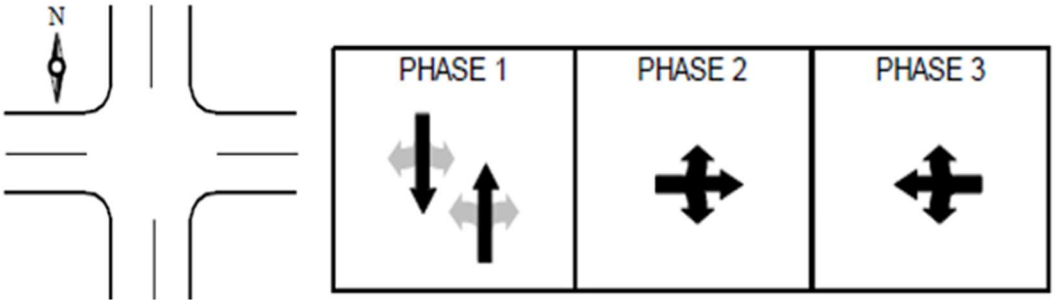

Figure 2.4. Three-phase signal movement (MnDOT, 2017)

2.3. Type of Traffic Control Strategies

According to Davol, A.P. (2001) and Robert, L., Gordon, P.E. and Warren-Tighe, P.E. (2005), traffic control systems may be classified in two main categories. The first category defines the manner that controllers react or operate according to traffic states, and it includes control strategies such as fixed-time, dynamic or adaptive. The second defines application areas of strategy, and it involves control strategies such as isolated, arterial coordination, and network.

2.3.1. Fixed-time traffic control

Fixed-time traffic control is the most widely used in the world. It is applied to isolated intersections as well as to traffic arterials and networks. The design of this system requires the determination of certain variables such as the number and sequence of phases, and phase lengths, which include green interval and yellow change interval. In addition, this determination is often based on traffic historical data (Robert, L., Gordon, P.E. and Warren-Tighe, P.E. 2005). In this system, the variables have fixed and predefined values, and the controller operates without taking into account traffic demands and conditions. As result, high traffic congestion can be observed during peak hours (Davol, A.P. 2001).

2.3.2. Adaptive traffic control

In contrast to fixed-time control, Adaptive Traffic Control (ATC) is a part of Intelligent Traffic Control System (ITCS) for Urban Transport System (UTS). It is applied to isolated intersections as well as to traffic arterials and networks. In this system, phase movements are determined according to traffic demands detected by sensors located at intersections. Also, cycle length and green time assigned to the phases are computed according to real time data detected and collected. In other words, this system responds automatically to the dynamic demands of traffic intersections or networks in real time, resulting in the reduction of traffic congestion (Aslania, M., Mesgari, M.S. and Wieringb, M. 2017). The development and implementation of this system is becoming more and more widespread around the world due to the inefficiency observed on the fixed-time control system. Furthermore, ATCS includes fully actuated and semi actuated traffic control.

2.3.2.1. Semi-actuated control

In this system, sensors are placed on minor roads as shown in Figure 2.5., and the controller allocates a minimum green time to these roads if the sensors detect traffic demands. Also, the controller continuously allocates maximum green signal to the major roads until a

demand is detected from the minor roads. This control is often used at intersections that only have high traffic demands on the major roads. The phase length and sequence depend on the traffic demand detected (Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. 2011).

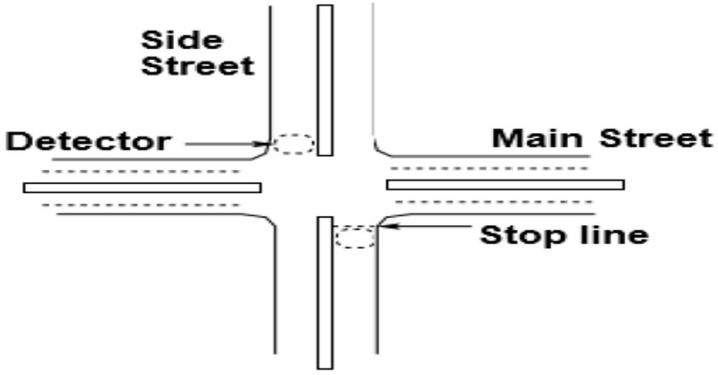

Figure 2.5. Sensors location for semi-actuated control (Badjie, O., Khalid, O. and Ali, O.M. 2016)

2.3.2.2. Fully actuated control

In contrast to semi-actuated control, fully actuated control consists of sensors placed on the major and minor roads as shown in Figure 2.6. (Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. 2011). Effective green time, phase length and sequence are determined according to the detection of traffic demands. Moreover, the first phase movement is assigned to the road with highest traffic demand and the last phase is assigned to the road with the lowest traffic demand.

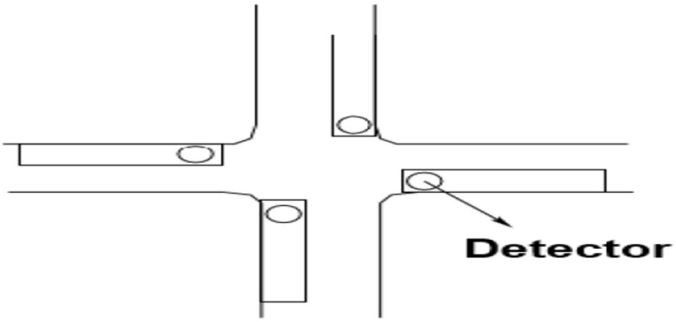

Figure 2.6. Sensors location for semi-actuated control (Badjie, O., Khalid, O. and Ali, O.M. 2016)

2.3.2.3. Green time calculation

According to Badjie, O., Khalid, O. and Ali, O.M. (2016), green time and cycle length in actuated control system may be calculated as follows:

$$G_{\min} = tL + 2n \tag{2.1}$$

Where:

- $G_{\min}$ = Minimum green time (sec),
- tL = Start-up lost time (sec), and
- n = Number of vehicles stored in the detection area.

Passage time interval may be computed as follows:

$$P = \frac{d}{s} \tag{2.2}$$

Where:

- P = Passage time interval (sec),

- d = Distance from detector to stop line (m/s), and
- s = Approach speed of vehicles (m/s).

Maximum green time may be calculated as follows:

$$g_i\left[C_i - L\right] + \frac{vci}{vc} \tag{2.3}$$

Where:

- g_i = Effective green time for Phase i (sec),
- C_i = Distance from detector to stop line (m/s),
- vci = Critical lane volume for Phase i (veh/hr),
- vc = Sum of critical lane volumes (veh/hr),
- L = Total lost time (sec).

According to Sharma, I. and Gupta, P.K. (2015), the maximum green time ($gmax$) allocated to the major roads may be calculated by expanding the cycle length applied within the minimum green time to 150 percent of its value.

Figure 2.7. shows how Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. (2011) summarized different intervals that include a duration of an actuated phase. First, the sensor detects a demand, the minimum green time is allocated to a phase, and the timer starts to count down. Then, when zero is reached, the expansion of this green time starts until the passage time timer reaches zero and resets, which leads the phase to finish with a gap-out. Finally, the timer counting down restart when the sensor detects a new demand.

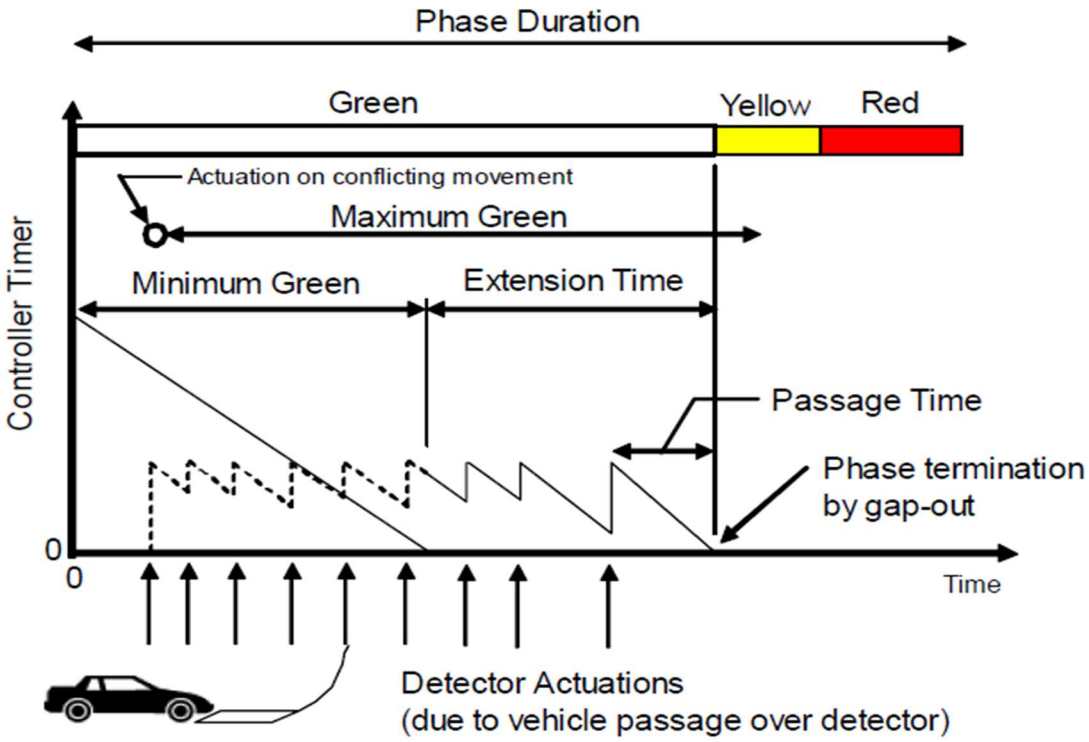

Figure 2.7. Intervals of actuated phase duration
(Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. 2011)

2.3.2.4. Cycle length calculation

Al-Mudhaffar, A. (2006) and NCHRP (2015) presented the Webster's model, which permits to calculate an optimal cycle by taking into account traffic constraints. This model is based on critical intersection method, and may be applied in isolated control as well as in coordinated control since it permits to reduce traffic delay at intersections, and should be in the range of 0.75Co and 1.5Co. The formula defined by this model is described as follows:

$$C_{Opt} = \frac{aL + b}{1 - y_1 - y_2 - y_3 \ldots y_n} \tag{2.4}$$

$$C_{Opt} = \frac{aL + b}{1 - \sum_{i=1}^{n} y} \quad (2.5)$$

Where:

- C_{Opt} = Optimal cycle length (sec),
- $y_1, y_2 \ldots y_n$ = Maximum ratios of flow to saturation flow for phases 1,2….n,
- $Y = \sum_{i=1}^{n} y$ = Sum of flow ratio, and
- a, b = Constants.

Al-Mudhaffar, A. (2006) said that the variables *a* and *b* should take the following values 1.5 and 5 for some balance in traffic flows, thus the optimal cycle length can be written as follows:

$$C_{Opt} = \frac{1.5L + 5}{1 - Y} \quad (2.6)$$

2.3.3. Isolated traffic control

Isolated control system is a control in which controller operates by taking local conditions of the traffic of a single intersection without considering adjacent signals (from other intersections). These conditions generally include capacity of intersection, delay of vehicles, safe and orderly traffic movement (Robert, L., Gordon, P.E. and Warren-Tighe, P.E. 2005). As mentioned before, fixed-time control, and adaptive control strategies may be applied within this control. In case wherein adaptive control is applied, sensors should be used for the purpose of traffic demand detection.

According to Al-Mudhaffar, A. (2006) the formula (2.6) may also be used to compute the cycle length, and equality of the ratio of effective green time (g_i) and the flow ratio (y_n) values should allow to compute the effective green times of phases.

$$\frac{g_1}{g_2} = \frac{y_1}{y_2} \ldots \tag{2.7}$$

With g_1 and g_2 as the effective green times of phases 1 and 2, and if $C_{Opt} - L$ is the total effective green time within the optimal cycle length, effective green times can be computed as follows:

$$g_1 = \left[C_{Opt} - L \right] \left(\frac{y_1}{\sum_{i=1}^{n} y} \right) \tag{2.8}$$

$$g_2 = \left[C_{Opt} - L \right] \left(\frac{y_2}{\sum_{i=1}^{n} y} \right) \text{ and so on} \tag{2.9}$$

2.3.4. Coordinated traffic control

In this control, controller operates by considering adjacent signals. This implies coordinated arterial and network control since the traffic policies used in an arterial network may simply be extended in order to apply them within a network (Davol, A.P. 2001). Before designing a coordinated control, designers or engineers need to think seriously about some conditions related to environment, traffic, and to motorists. These conditions are often affected by parameters such as congestion, road traffic volumes, user safety, arterial speeds, cycle length, and green signal duration. For example, close intersections with a high volume of traffic need an implementation of coordinated controls because they probably offer more benefits. In contrast to close intersections, very distant intersections first require a study of traffic volumes before deciding on the implementation of a coordinated control between them. All of these considerations make the design more difficult (NCHRP, 2015).

2.3.4.1. Coordination objective

The main objective of coordinated control system is to establish a green wave in which vehicles can move from one intersection to another without stopping, which effectively increase the vehicle numbers that pass along a traffic arterial or network. This contributes to the reduction of travel time, stops, delay, and long queues. According to Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. (2011), this objective can be reached by designing an algorithm that is capable to:

- Adjust traffic parameters in real time to maintain coordination operations,
- Distribute the same cycle length to all controllers at the same time, and
- Define phase intervals in this cycle length.

2.3.4.2. Coordination timing plan

According to Robert, L., Gordon, P.E. and Warren-Tighe, P.E. (2005), practically a coordination timing plan guidelines include, (1) coordinated phase assignment, (2) cycle length selection, (4) split distribution, and (5) offset optimization.

2.3.4.3. Master clock importance

Master clock is the brain of coordinated control system. It is within it that the control logic is established to which all other controllers (local controllers) of system refer in their operations (NCHRP, 2015). Figure 2.8. presents the organization of the operations of master controller (master clock) and local controllers (local clocks) under a coordinated phase, which is referred to the phase 2. In this figure, the master clock is located on the horizontal axis, and the local clock next to each intersection. Also, the green waves established by this coordinated phase are represented by both blue and purple perpendicular lines.

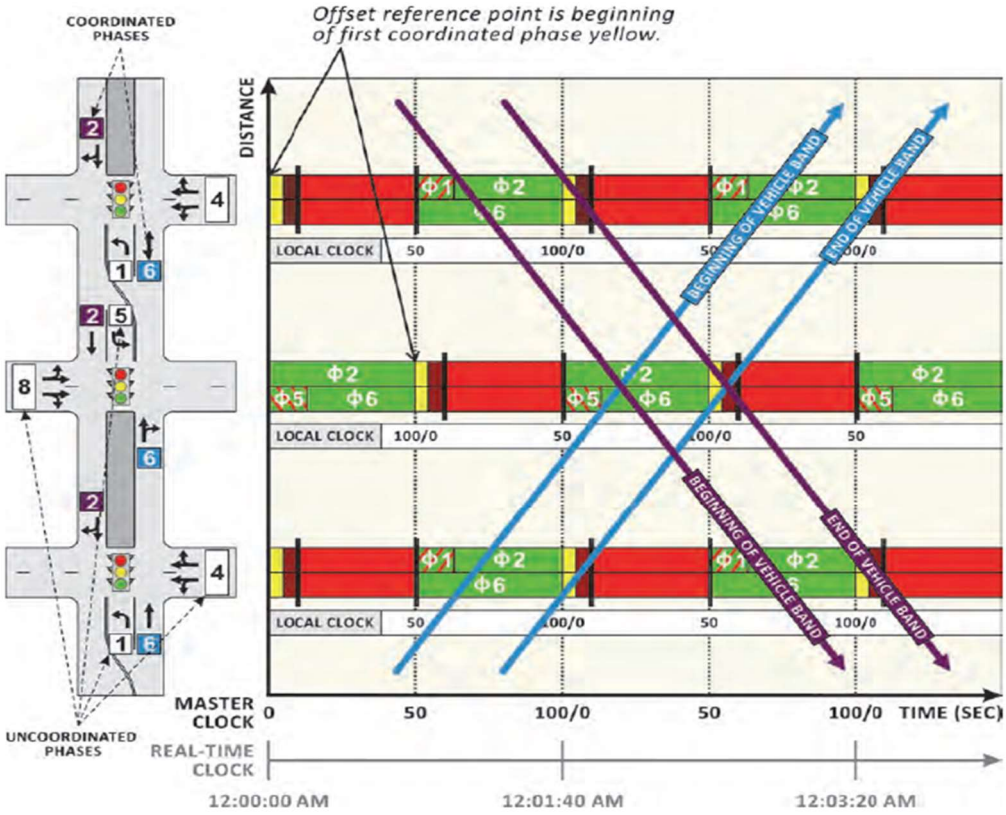

Figure 2.8. Master and local clock organization (NCHRP, 2015)

2.3.4.4. Effective green time and phase length calculation

Several methods allow to compute effective green time and phase length among which MnDOT (2017) proposed the following formula:

$$G_i = G_t\left(\frac{vi}{V}\right) \tag{2.10}$$

$$G_t = C_{Opt} - (yell_1 + yell_2 + ...yell_n) - tL \tag{2.11}$$

$$V = \sum_{i=1}^{n} vi \tag{2.12}$$

Where:

- G_i = Green time for phase i (sec),
- G_t = Net green time or split green (sec),
- $yell_1, yell_2 ... yell_n$ = Yellow time for phases 1,2….n,
- V = Sum of critical line volume (vph), and
- n = Number of phases, and
- tL = Lost time in phase i.

$$\text{Phase duration}_n = G_i + yell_n + tL \tag{2.13}$$

The sum of Phase n should be equal to optimal cycle length (MnDOT, 2017).

2.4. Works Related to Control Systems

Several researchers, engineers, students and organizations have devoted their efforts to the design and implementation of control strategies as well as to the projects based on the control of traffic lights. Thus, we conducted deep and serious studies of more than 100 of these project and strategy, from which we selected the most relevant to describe in this thesis.

2.4.1. Old control strategies

Old control strategies are considered as the pioneers of control strategies, which inspire developers, engineers, and student to develop new traffic control strategies. Among old strategies studied, the most relevant are present as follows.

The Sydney Coordinated Adaptive Traffic System (SCATS) is a control strategy developed in 1970 by the Department of Main Roads of New South Wales in Australia (Al-Mudhaffar, A. 2006). The control logic included in this strategy allows it to respond in real time to dynamic demands of traffic detected by sensors (Robert, L., Gordon, P.E. and Warren-Tighe, P.E. 2005). Its control is based on a strategic control that establishes a timing plan according to the traffic conditions of network. In addition, it is based on a tactical control that provides a local control of network intersections (Shelby, and Gebhart, S. 2001).

The Split Cycle and Offset Optimizing Technique (SCOOT) is one of the World's foremost adaptive control strategy developed in the early 1973's, by the Transport and Road Research Laboratory (TRRL) in Great Britain (Robert, L., Gordon, P.E. and Warren-Tighe, P.E. 2005; Shelby and Gebhart, S. 2001). This strategy uses traffic data collected at intersections in order to compute and to optimize splits, cycle time, and offsets applied to the entire network. Moreover, the time plans established within this strategy are continually updated and adjusted either offline or online. This is performed in accordance with TRANSYT's control strategy (Al-Mudhaffar, A. 2006; Shelby and Gebhart, S. 2001).

The Real-time Hierarchical Optimized Distributed Effective System (RHODES) is an adaptive traffic control strategy developed by University of Arizona researchers in USA. This strategy aims to optimize traffic flows at many stages. It is based on the activated vehicle's call logic, which uses upstream and downstream sensors. Each upstream sensor sends vehicle numbers waiting in queue when its predefined threshold is reached. The downstream sensors are used to predict the arrival of vehicles (Negi, P. 2006).

Transit-Oriented Development (TOD) is one of the oldest automatic fixed-time control strategy that was developed in the early 1922's (Thorne-Lyman, A., Jeff Wood, J., and Sam Zimbabwe, S. 2000). Its control logic is based on the predefined timing plans that can be chosen to operate at a specific time, i.e., time of day, day of week, week of year and special holiday (MnDOT, 2017; Thorne-Lyman, A., Jeff Wood, J., and Sam Zimbabwe, S. 2000). The parameters that may often change from one plan to another include, cycle length, green times, phase duration, and phase sequence (MnDOT, 2017). Once timing is chosen, all parameters are maintained constant. Furthermore, this strategy involves a

predicted timing plan established using the past traffic data obtained from the traffic measurements.

Nathan Gartner developed optimization Policies for Adaptive Control (OPAC) in the 1980's in order to reduce traffic delay and travel time at intersections or in networks (Shelby and Gebhart, S. 2001). This strategy is based on dynamic demand-responsive logic related to the dynamic programming (DP) formulation. It includes online application for computing traffic parameters and for defining the timing plans. Besides, this strategy was designed in different versions such as OPAC-II, OPAC-III, OPAC-IV, and OPAC-RT, which provide the "Projection Horizon" process. With the help of this process, it is possible to compute a new cycle length each time traffic data is updated (Robert, L., Gordon, P.E. and Warren-Tighe, P.E. 2005).

The Urban Traffic Optimization by Integrated Automation (UTOPIA) is a hierarchical decentralized control strategy developed by Italian researchers (Shelby and Gebhart, S. 2001). This strategy firstly uses a global control of traffic network in which coordination and timing plan are determined according to the traffic conditions of the entire traffic network. It secondly uses a local control of a intersection named Decentralized Optimization System (SPOT), which is an adaptive control that responds in real-time to the dynamic demands of an intersection (Al-Mudhaffar, A. 2006).

For other old strategies, details on MAXBAND, TRANSIT, and MITROP are available in (Shelby and Gebhart, S. 2001). Also, information about LHOVRA and UTOPIA control strategies can be found in (Al-Mudhaffar, A. 2006).

2.4.2. Recent control strategies

Aslania, M., Mesgaria, M.S. and Wieringb, M. (2018) developed an adaptive traffic signal control with actor-critic methods in a real world traffic network with different traffic disruption events. In the end to reduce traffic congestion, they proposed a Reinforcement learning (RL) algorithm named actor-critic algorithm that predict future traffic conditions and demands at intersections. The operations of this algorithm are based on two main

agents. The first is a critic agent that analyses current traffic conditions in order to determine better or worse condition. The second is an action agent that reacts according to traffic conditions found by the first agent. Moreover, the first agent predicts future traffic conditions with the help of a temporal difference (TD) error that is based on previous and current traffic condition assessment. The proposed adaptive algorithm was tested within a controller, which was benchmarked against controllers using Bayesian Q-learning, fixed time and actuated algorithm. The test results showed very high performances for this adaptive algorithm.

Mfenjou, M.L, Abba-Ari, A.A., Abdou, W., Kolyanga and Spies, F. (2018) developed a methodology and trends for an intelligent transport system in developing countries. Their study aims the reduction of traffic accidents and traffic congestion within traffic networks. Thus, their model proposed an algorithm automatically computes the dynamism of traffic roads in order to suggest to the motorists free roads in case of traffic congestion on a part of traffic network. The traffic policy included within this algorithm was defined to provide some functions such as Self-Configuration, Self-Optimization, Self-Healing and Self-Protection. All of these functions play important roles in terms of, (i) real time traffic data acquisition, (ii) the update of traffic data, and (iii) their improvement. The result tests prove that this model efficiently regulates and manages traffic by reducing long queue at intersections.

Gao, J., Shen, Y., Liu, J., Ito, M. and Norio, S. (2017) proposed an Adaptive Traffic Signal Control (ATSC) based on deep Reinforcement Learning (RL) algorithm with experience replay and target network. This algorithm is based on the machine-crafted features that provide possibilities to obtain traffic information in real time. The control logic learns traffic states thanks to an agent that interacts with an intersection, by (1) observing traffic demands in real time, (2) by performing demand comparison, and (3) by allocating green signal to required road lines. Once green line is allocated, this agent gets a feedback that allows to assess efficiency of its taken decision. The necessary input information for this agent includes the position and speed of vehicles. Experimental test results showed the vehicle delay decrease in the range of 47% to 86%.

Sayyed, S., Date, P., Gautam, R. and Bhandari, G. (2014) proposed a Design of Dynamic Traffic Signal Control System (DTSC). This system targets to control efficiently traffic flow in order to increase the rate of safe in UTS. Control strategy developed in this system provide perpetual measures of traffic density during peak time of the days. Besides, mathematical computing are used to determine green time, splits and cycle length according to input traffic data (queue of vehicles) collected at intersection. Green signal is allocates to roads lines in ascending order determined by theses mathematical computing.

Kouvelas, A., Aboudolas, K., Papageorgiou, M., and Elias B.K. (2014) developed a Hybrid Strategy for Real-time Traffic Signal Control (TSC) of Urban Road Networks (URNs). This strategy implies two main control logic. The first is a Demand-Based (DB) signal control that allows each intersection controller to compute green time according to traffic congestion detected. The second is a format of Linear-Quadratic (LQ) regulator, which detects the saturation of traffic and activates the DB control logic. According to the obtained result tests, this hybrid strategy provided an efficiency of approach during either the off-peak time or peak time.

Srinivasan, D., Choy, M.C. and Cheu, R.L. (2006) developed the neural networks for Real-Time Traffic Signal Control (RTSC). An algorithm consists of two multi-agent systems was proposed to control locally more than a traffic intersection at the same time. One of these multi-agent systems is based on the calculation of hybrid smart process, which continuously updates traffic data and make decision to responds to traffic demands. Another multi-agent system is based on the simultaneous perturbation stochastic approximation logic, applied within fuzzy neural networks (NN). It authorizes the execution of operations carried out by the first system, and performs adjustments in case of loss of data caused by disturbances during data analysis. Test results demonstrated an effectiveness from 78% to 85% in terms of vehicle delay and stop time reductions.

Mishra, S., Bhattacharya, D. and Gupta, A. (2018) proposed congestion adaptive traffic light control and notification architecture using Google maps APIs (Application Program Interfaces). The principle goal of this system is either to reduce or to avoid road traffic congestion. To this end, its adaptive algorithm is based on a process of acquiring traffic

data on a website. This permits to analyze the most crowed (intensive) road line data that are extracted from Google page. After this data extraction, green time and cycle length computation are performed in order to be assigned to phase movements. The simulation results showed a short response time to the different variations of traffic conditions. They also confirmed the effectiveness of the system in its implementation on real traffic network.

Aslania, M., Mesgari, M.S. and Wieringb, M. (2017) described a dynamic green split optimization designed for urban street network. The control strategy developed in this system considers a part of traffic network from which it updates traffic data and performs analyses taking into account the state of the entire traffic network. This method leads the system to achieve congestion reduction for all intersections of urban traffic network. The results suggested that it is possible to reduce traffic delay by approximately 35% by applying this method during AM and PM peak hours of day.

2.4.3. Traffic control systems based on the Arduino Boards

Valhavankar, S.N., Vibhute A.S., Bhagat, A.G., and Said S.K. (2016) developed an intelligent traffic control system based on emergency vehicle clearance and detection of the lost vehicle. This system consists of the following components:

- Traffic lights located at each line of intersection;
- RFID (Radio Frequency Identification) readers placed at each side of the intersection to identify vehicles;
- Arduino UNO, that controls the traffic system and counts the number of the vehicle passing through the intersection thanks to the IR sensors placed at each side of the intersection; and
- Server that receives and stores vehicle information coming from the RFID readers, and sends the information to the police office via an android application if necessary.

The Vehicles passing through the intersection are equipped with a radio frequency identification (RFID) tag readable by the RFID readers. Each RFID carries a specific

number allowing the identification of the vehicles. When a vehicle breaks the traffic control rules, the identification process first detects it. Then, sends its identification number to the nearest police station from where it is located in order to intercept it. Finally, a fine is taken from the bank account associated with its identification number. In addition, if it is a lost or stolen vehicle, this system allow to locate, track and to find it. When an emergency vehicle reaches the intersection, this identification process gives it a priority to pass without wait.

Lalitha, K. and Pounambal, M. (2018) developed the density based traffic management system using IoT. This system consists of traffic lights located at intersection, the Arduino Mega 2560 controller located in a controller box controlling the traffic signal, IR sensors placed at each side of the intersection for the vehicle counting, and an Arduino Ethernet shield that establishes a communication between the Arduino controller and a website. Firstly, the IR sensors detect and count the number of vehicle passing through the intersection, and send the information to the controller. Then, the controller performs a density comparison in order to determine the most intensive line. Once this intensive line is determined and selected, the controller send this information to the website database that updates the traffic every time for traffic adjustment purpose. Finally, the green signal of the first phase movement is allocates to the most intensive line with sufficient time duration allowing all vehicles to pass through the intersection. The algorithm proposed by this system permits the controller to react continuously and dynamically to the traffic density in order to avoid congestion at the intersection.

Varun, K.S., Kumar K.A., Chowdary, V.R. and Raju C.S.K. (2018) proposed a perceptive model of traffic flow using Arduino board for reducing the traffic congestion at an isolated traffic intersection. This is a density-based traffic management system using a method of measuring the weight of the vehicle sensed at each line of an intersection. For this goal, the system consists of the following components:

- Arduino UNO used as the main controller;
- Traffic lights (LEDs) placed at each line;

- Load cells located at each side of the intersection;
- HX711 sensors placed at each side of the intersection, wired to the controller and to the load cells; and
- Liquid crystal display (LCD) wired to the controller.

Once the vehicles are sensed by the load cells, they send the signals to the HX711 sensors that process, amplify and converts them into the vehicle weight values. Then, the controller reads those values that it sends to the LCD for displaying. Finally, the line with the highest weight is considered as a more intensive line, therefore, its green light turns ON, and the red lights of other lines turn ON until the weight value of this intensive line decreases reaching its initial value. The weight of vehicles is detected, and converted into density in a continuous manner, thus ensuring an adaptive traffic control system. According to the test results, this system reduces traffic congestion at acceptable rate. Also, it is less expensive than a system using the RFID sensors.

Rajavali, G., Sravani, Y., Raju, P., Mrudhulatha, G. and Appalnaidu, CH. (2017) developed an Arduino based smart roads controlling system for future cities. Indeed, the system aims to reduce traffic congestion, delays, and traffic system power consumption by controlling the traffic lights as well as the streetlights. The traffic is controlled by applying the method of the vehicle density measurement. This control is made possible with the help of such components, the Arduino Uno controller, Infrared sensors, light dependent resistors (LDRs), and light emitting diodes (LEDs). The control procedure involves a continuous detection and measurement of density by the IR sensors, green light assignment according to density volume read by the Arduino Uno controller, and the turning ON of the streetlights by the LDRs which is only performed at night when a vehicle is detected. The system is developed in a way that allows the controller to first allocate green signal to the line with the highest density, and last to the one having the lowest density.

Arifin, M.F.B. (2013) developed a solar powered smart traffic light system that aims to reduce traffic congestion and energy consumption by using the IoT technologies. For these ends, it includes an Arduino board controller, GPS modules, XBee modules (transmitter / receiver), intelligent detectors as well as solar panels. The technique used in this system is

which that counts the number of vehicles detected at each line. The controller performs a comparison of those vehicle's numbers, and turns ON the green light of the line having the highest vehicle's number in first position. The green light of the line having the lowest vehicle's number is turned ON in the last position. Also, if an emergency vehicle is detected, it gets an ON green light without waiting (priority case). Each emergency vehicle is equipped with a GPS module that locates its position, and an XBee transmitter that transmits this location to the nearest Arduino controller connected to an XBee receiver. This permits the controller to maintain the green light ON while emergency passing the intersection. In addition, a battery wired to a solar panel is used to charge all of these equipments (system devices).

Safi, M.E. (2016) developed a smart traffic light controller based on microcontroller, especially on an Arduino Uno controller, in order to reduce traffic congestions. The system provides the possibility to control the traffic signal in two ways. The first way consists of a number of ultrasonic sensors placed along each line, which are activated when the vehicle passes. Therefore, depending on the number of sensors activated on each line, the controller easily determines the most congested line, which receives a green signal with sufficient time duration. The second way includes switches providing manual control of traffic system by traffic police officers. Also, these switches enable to activate the automatic or manual mode.

Ashok, P.V., Siva, S.S., Vignesh, M., and Sankaranarayanan., S. (2017) proposed an IoT based traffic signaling system due to the limitations presented by the control system based on fuzzy logic. This system consists of an Arduino Uno controller, ultrasonic sensors, Raspberry Pi3, and a Wi-Fi module. First, the ultrasonic sensors located at every 50 meters of each line measure the traffic density and send this information to the controller. Then, the controller determines the most intensive line, and allocates it the green signal with a long duration depending on the density volume. Also, the controller sends this density information to the Raspberry Pi3 working with the Wi-Fi module for their processing. Finally, the density information including the date and time of each line is transmitted to a webpage for possible analyses from traffic police officers.

2.5. Conclusion

In this chapter, a brief history of traffic control systems was presented. The basics of traffic control system was deeply discussed. Finally, some old and recent works related to traffic control strategy and project were described.

This chapter shows that more efforts have been dedicated to adaptive traffic controls based on isolated intersections than on coordinated arterials or networks. This leads to a huge lack of documentation presenting practical implementations of dynamic control systems applied on coordinated networks. In addition, several works focused on theoretical details than practical. Thus, to fill this gap, this thesis focused on design and implement of an adaptive traffic control system, which is dynamically coordinated.

3. ARDUINO BOARD OVERVIEW

3.1. Introduction

Before, coding and electronics were left to certified technicians and engineers since most of people thought that these were difficult to learn. Nowadays, thanks to Arduino boards, everyone can easily learn how electronics components operate, and how to control them by writing codes with a simplest version of C++ programming language within Arduino software. Several projects related to the new technology include these boards as masters controllers (Fitzgerald, S. and Shiloh, M. 2012; Baichtal, J. 2014).

This chapter presents an overview of the Arduino boards, in which some topics will be covered, such as, (i) different Arduino boards, (ii) how to choose an Arduino board (iii) the Integrated Development Environment (IDE), and (iv) programming Arduino Boards.

3.2. Definition of the Arduino Boards

Arduino board may be defined as a smart small computer that can be connected to electrical, electronic and electromechanical devices in order to control or command them.

The interaction between Arduino and external devices is made possible with the help of a microcontroller, which is its brain. This microcontroller generally consist of electronic circuits, which includes a memory part, and input/output parts (Smith, A., G. 2011; Anonym, 2016). Arduino board attracts artists, teachers, students, hobbyists, and anyone interested in artificial intelligence, because it is an open-source platform and extensible

hardware. Furthermore, it has a simple programming environment to use and an extensible software (Smith, A., G. 2011).

3.3. Different Arduino boards

There are several types of Arduino boards available, including boards and shields. The difference between these types depends on the microcontroller performances, number of inputs and outputs, operating voltage, speed, dimension, and on other parameters (Shah, S. and Shah, U. 2016; Anonym, 2016). Table 3.1. presents a summary of the Arduino boards.

Table 3.1. Summary of Arduino board types (Anonym, 2016)

Name	Processor	Clock Speed	Operating Voltage	Memory	Digital I/O(PWM) pins	Analog input pins
Arduino UNO	ATmega328P	16 MHz	5 V	2KB SRAM, 32KB flash	14(6)	6
Arduino Mega	ATmega2560	16 MHz	5 v	8KB SRAM, 256KB flash	54(15)	16
Arduino YUN	Atmega32U4	400 MHz	4.75 v	64MB SRAM, 16MB Flash	14(6)	12
Arduino Due	ATSAM3X8E	84 MHz	3.3 v	96KB SRAM, 512KB flash	54(12)	12
Arduino Lilypad	ATmega32u4	8 MHz	3.3 v	2.5KB SRAM, 32KB Flash	14(6)	6
Arduino Zero	ATSAMD21G18A	48 MHz	3.3 v	32 KB SRAM, 256 KB Flash	14(6)	6
Arduino Mini	ATmega328	8 MHz	3.3V	2 KB SRAM, 32KB Flash	14(6)	6

Figure 3.1. illustrates the images of these Arduino boards, and more detail about them can be found in (Schwartz, M. 2014; Anonym, 2016; Shah, S. and Shah, U. 2016).

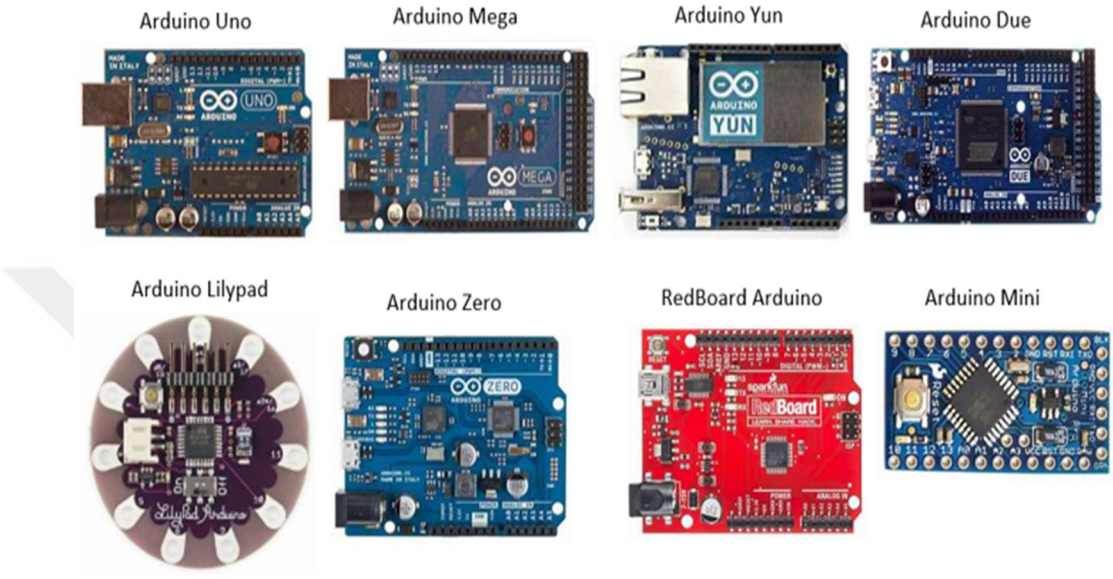

Figure 3.1. Images of Arduino board types (Shah, S. and Shah, U. 2016)

3.4. How to Choose an Arduino Board

Choosing an appropriate Arduino board for a project seems to be a difficult task because of the large variety of Arduino boards, and no one can know all technical specifics from all of them. Thus, Shah, S. and Shah, U. (2016) proposed a simple flowchart facilitating the choice.

From this flowchart, the following question should be answered. Have you decided on the Arduino Project? If the answer is no, the Arduino UNO should be chosen because of its compatibility with the most existing libraries and shields. If the answer is yes, the next question should also be answered. In which area? Thus, Arduino board will be chosen according to the project area as shown below (in Figure 3.2.).

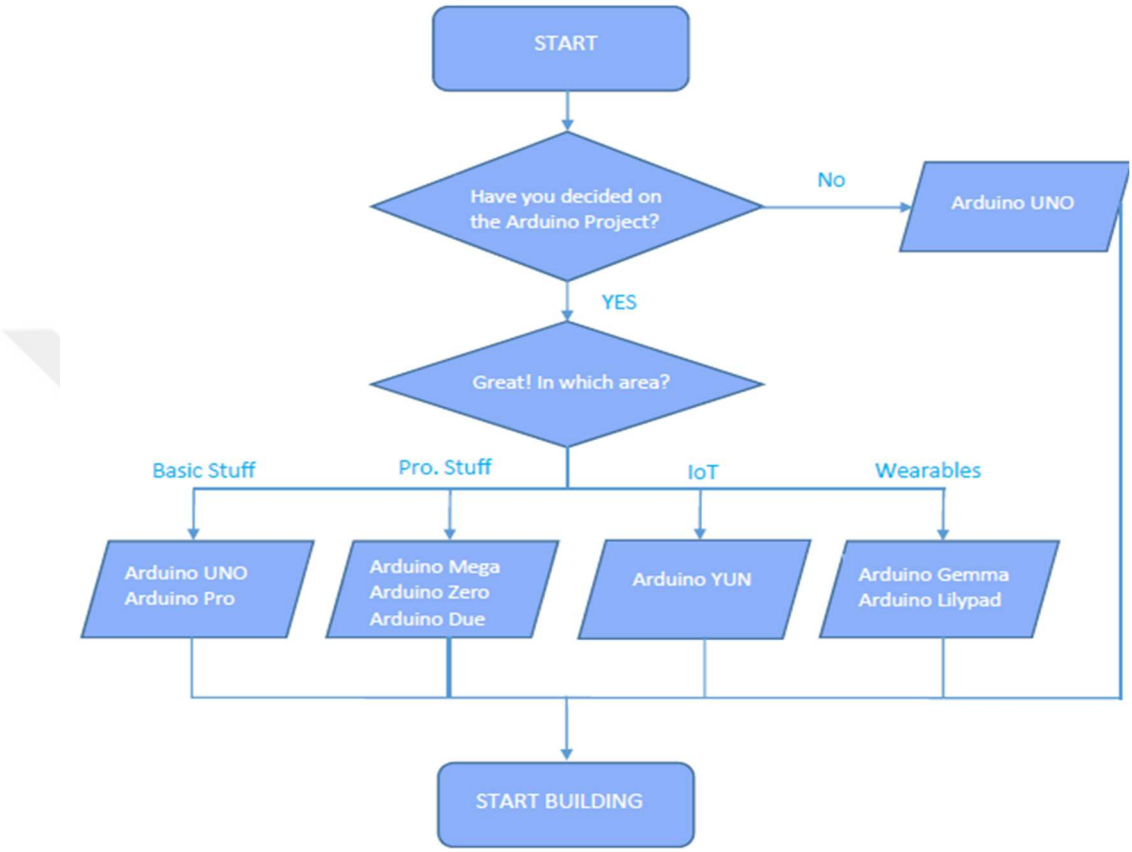

Figure 3.2. Flowchart for Arduino boards choice (Shah, S. and Shah, U. 2016)

3.5. Arduino Boards for this Thesis

Once we have answered to the flowchart questions, we chosen the Arduino boards Mega and UNO for our thesis. Their technical specifications were previously presented (in Table 3.1.).

3.5.1. Arduino UNO

The Arduino UNO is the most used in the world among Arduino board types. It is based on the ATmega328P microcontroller developed by Atmel company, and contains components similar to those of most boards. Its name (UNO) come from Italian language, which means one indicating the first USB Arduino board version. Its microcontroller (ATmega328P) consists of processor, memories in which codes can be stored, and programmable input-output pins (Smith, A., G. 2011; Anonym, 2011). The relevant components available on this Arduino board are shown below (Figure 3.3.).

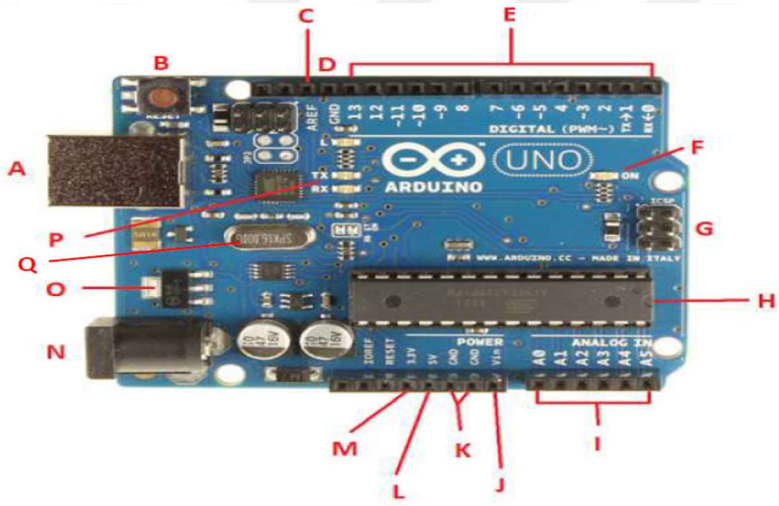

Figure 3.3. Components of the Arduino Uno (Shah, S. and Shah, U. 2016)

- A: USB plug, which allows to power the Arduino board via a USB connector, and to upload the program to the computer (Fitzgerald, S. and Shiloh, M. 2012).
- B: Reset button, which allows to reset the ATmega328P microcontroller leading to start the execution of the uploaded code at the beginning (Anonym, 2012).
- C: AREF pin, which allows to establish a voltage limit in the range of 0 to 5 volts for the analog, pins (Anonym, 2012).
- D, K: Ground (GND) pin that allows to put a circuit to zero or negative voltage.
- E: Digital pins (0 to 13) can be used as inputs for reading input values from external components, i.e. pushbuttons, and as outputs for sending commands to components,

i.e., turn on or off the lights. Among these pins, there are those that can be used to provide a PWM output, i.e. pins 3, 5, 6, 9, 10 and 11. In addition, the pins 0 (RX) and 1 (TX) are used for serial communication between Arduino boards (Shah, S. and Shah, U. 2016; Anonym, 2012).

- F: Power LED indicator permits to indicate that the Arduino board is powered properly by flashing, or not by remaining off during the power-up process (Shah, S. and Shah, U. 2016; Anonym, 2012).
- G: In-circuit serial programmer (ICSP) pin offering possibilities for programming the Arduino board using another Arduino board (Shah, S. and Shah, U. 2016; Anonym, 2012).
- H: Integrated Circuit (IC) is an ATmega328P microcontroller developed by Atmel Company. Also, it is the most important component, which stores the program, and manages all operations of Arduino UNO.
- I: Analog pins (A0 to A5) used to read input values coming from analog detectors such as temperature sensors, and then convert them into digital values readable by the ATmega328 microprocessor (Anonym, 2012).
- J: Vin pin from which the Arduino board can be powered with a DC voltage of up to 12V (Shah, S. and Shah, U. 2016).
- L, M: pins that provide DC voltages (3 and 5 v) to power external components operating with these voltages (Shah, S. and Shah, U. 2016).
- N: External power supply, which provides another possibility to power up the Arduino board with an AC voltage of up 12 v (Fitzgerald, S. and Shiloh, M. 2012).
- O: Voltage regulator controls and regulates the voltages powering some electronic components located on the Arduino board (Anonym, 2012).
- P: TX and RX LEDs, which permits to indicate a serial communication between the Arduino board with the computer. They flash when the board receives or sends data (Fitzgerald, S. and Shiloh, M. 2012).

- Q: Crystal Oscillator, which allows the Arduino board to calculate the task execution times. This is expressed in Hertz (Frequency of execution) as shown in clock speed from the column of Table 3.1. (Anonym, 2012).

3.5.2. Arduino Mega

The Arduino Mega 2560 is based on an ATmega2560 microcontroller developed by Atmel company. This microcontroller includes a processor, memories in which codes can be stored, programmable I / O pins, and other similar components located on the Arduino UNO. Its physical components are shown below (in Figure 3.4.).

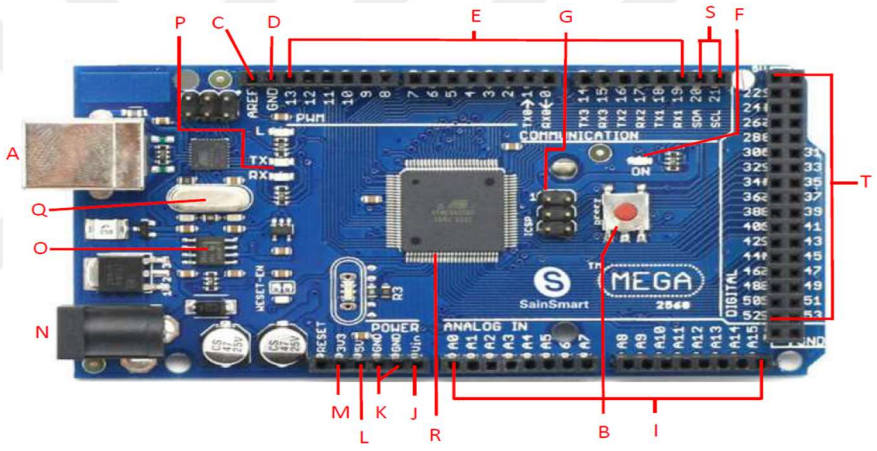

Figure 3.4. Components of the Arduino Mega

As mentioned before, the Arduino UNO and Mega have almost the same components, except those described below.

- E, T: Digital pins (0 to 52), whose eight can be used for serial communication. Serial 0 includes the pins 0 (RX) and 1 (TX), Serial 1 the pins 19 (RX) and 18 (TX), Serial 2 the pins 17 (RX) and 16 (TX), and Serial 3 the pins 15 (RX) and 14 (TX).
- R: ATmega2560 microcontroller, which is the brain or the heart of the Arduino Mega.
- S: SDA and SCL pins from which serial communication between Arduino boards can also be established with the help of libraries, i.e., the Wire library.

3.5.3. Integrated Development Environment (IDE)

The Arduino software (IDE) is a free software developed by the Arduino team for the programming of all Arduino boards. This software provides a clear and simple environment for writing code with the simplest version of C++ programming language. In addition, this software includes features that allow uploading programs (codes) to a computer, as well as to compile and complete them intelligently.

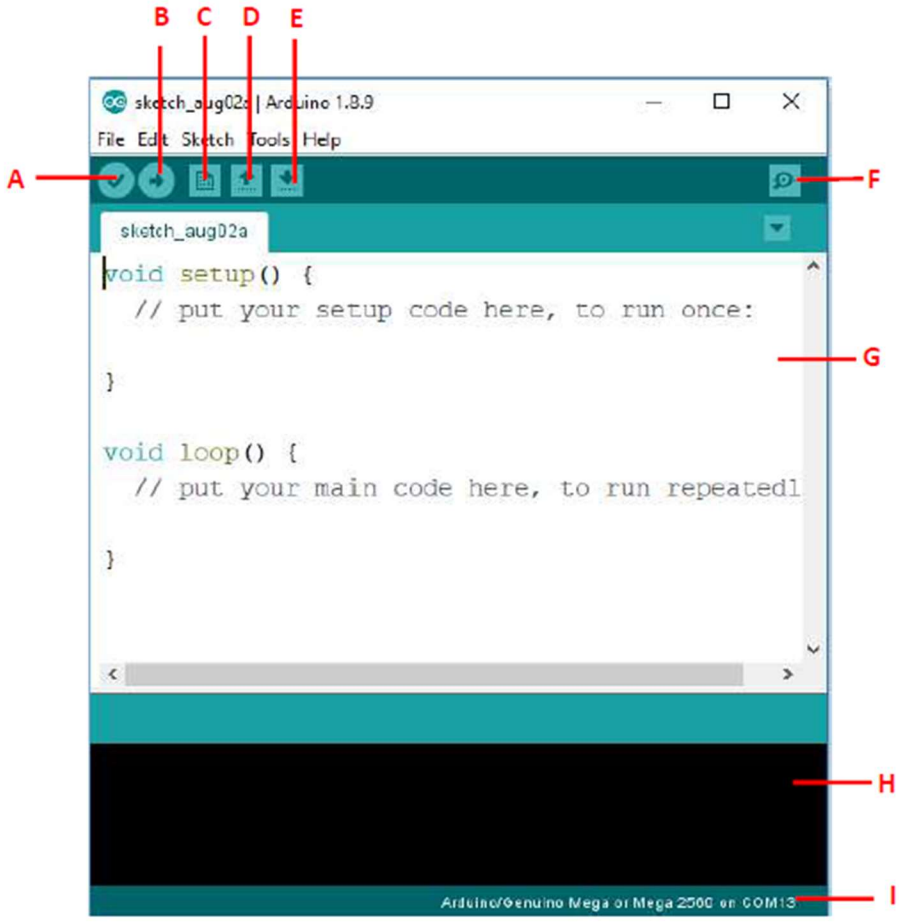

Figure 3.5. Menu items of the IDE (Shah, S. and Shah, U. 2016)

The description of menu items and buttons located on the Integrated Development Environment (IDE) software is shown below (in Table3.2.) as follows.

Table 3.2. The IDE software menu and button description

Menu items and Buttons		Definition
A	Verify button	Allows to compile code and verify code error
B	Upload button	Allows to upload code to the Arduino board
C	New button	Allows to open a new coding interface (new sketch)
D	Open button	Allows to open existing codes from computer
E	Save button	Allows to save code
F	Serial monitor	Displays the serial monitor screen
G	Editor window	Area in which code must be written
H	Error console	Area in which error message is displayed
I	Status bar	Displays the Arduino name and the port on which it is connected

3.5.4. Downloading and installing of the IDE software

The Arduino official website page provides several free downloadable versions of the IDE software (open-source), whose the IDE 1.8.9 is the latest. These versions run on operating systems such as Windows, Mac, and Linux.

Once downloaded, the installation process will depend on computer (laptop) operating system. For more information about this process, we recommend you to read some books indicated in reference (Shah, S. and Shah, U. 2016; Smith, A., G. 2011 and Anonym, 2012).

3.5.5. The IDE libraries

The Arduino libraries can be defined as folders having many files. The IDE software includes some pre-installed libraries located in the library folder. There are many varieties of libraries that can be freely downloaded, and easily added to this IDE software. These libraries provide support and functionality allowing the communication between the Arduino board and external devices such as liquid crystal display, Bluetooth wireless

module, RF module, and so on. For example, the libraries like wire library, and serial library can be used in order to establish the serial communication between Arduino boards.

3.5.6. Program structure of the IDE Software

As stated earlier, the IDE software can be programmed using the C++ programming language. This language includes functions, values (variables and constants), and structure. Furthermore, the structure consists of two functions that are, setup( ), and loop( ) depicted in Figure 3.5. Table 3.3. shows the summarize of descriptions related to these functions, values, and structure. However, depth information are available on the Arduino official website or in the books like (Shah, S. and Shah, U. 2016), (Smith, A., G. 2011), and (Anonym, 2012), see the references.

Table 3.3. The IDE software code elements

Elements of the IDE C++	Description
Structure	The elements of the IDE C++ code
setup()	Used to initialize variables, pin modes, etc.
loop()	Located below the setup(), it is used to write the main code of a project
Values	Data types and constants
HIGH | LOW	Indicate the state 1 (5v) or 0 (0v) for the digital pins
INPUT | OUTPUT	Indicate that pins are configured as a input or output
int	Integer variable (integer value)
float	Float variable (decimal value)
unsigned long	Used to convert a value to the unsigned long data type
Functions	Used to control the Arduino board and to perform calculation
digitalRead() | analogRead()	Used to read the values from digital/analog pins
digitalWrite() | digitalWrite()	Used to write HIGH or LOW value to digital/analog pins
pinMode()	Used to configures pins as input or output
Serial	Used to establish a communication between the Arduino board and a computer or other hardware

4. PROPOSED TRAFFIC CONTROL SYSTEM

4.1. System Structure Description

The proposed system consists of three intersections controlled by three controllers. One of them is the master and the other two are the slaves. Each intersection contains four roads with two lines each. In addition, the sensors are placed on each entry line of the intersection, and serial links are used for the communication between the controllers.

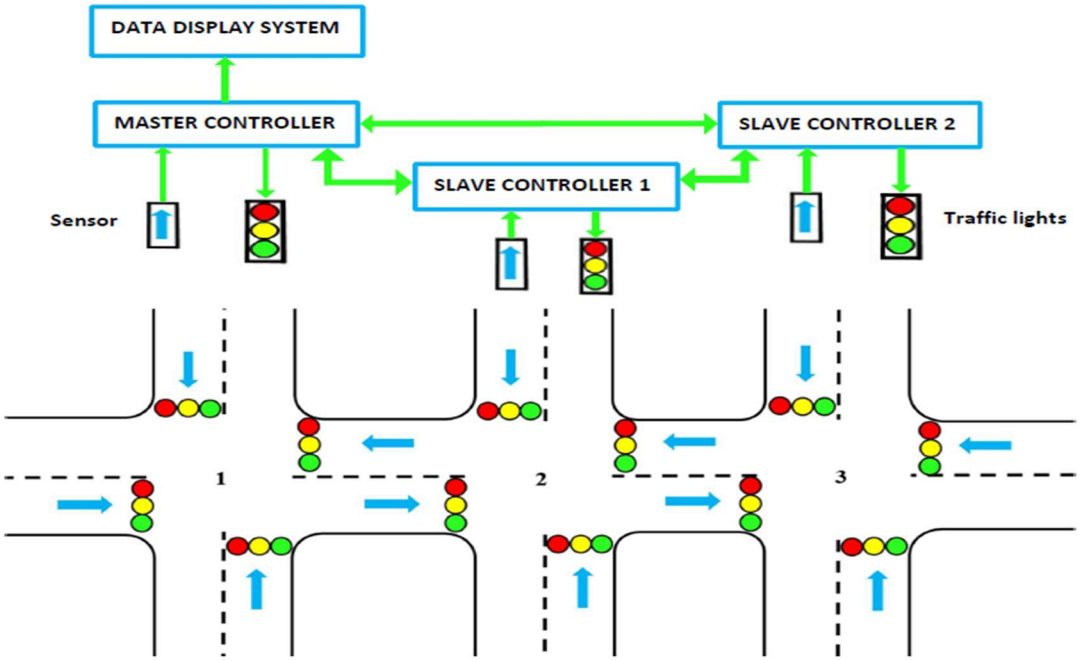

Figure 4.1. System structure

A general operating principle of this system can be explained as follows:

- Sensors detect and count vehicle numbers, and send this information to controllers,
- Each slave controller sends its detected vehicle numbers to the master controller,
- Master controller computes the cycle length value and sends this value to the slave controllers, and
- All controllers compute green time duration, compare the line traffic volume (vehicle numbers), decide what lines should be served first, and allocate the phase green time to the corresponding lines.

4.2. Algorithm Development

As stated early, one of the objectives of this thesis is to develop an adaptive control algorithm, which will reduce traffic congestion. Therefore, to minimize travel times, stops, and delay along entire traffic networks. To this end, the developed algorithm firstly includes a possibility to apply the optimum cycle length duration determined on critical intersection method. This allows to compute the traffic green light time. Moreover, contrary to some cycle length computing methods used, this critical intersection approach considers the intersection constraints such as the lost time and traffic saturation flow rate, which intends to minimize average delays and to maintain intersections under saturated conditions (MnDOT, 2017; NCHRP, 2015). Secondly, it allows to establish an appropriate coordination-timing plan in order to provide a free travel without stopping along of arterial network. Finally, it allows to distribute effective green duration according to the vehicle numbers detected on each road line.

The coordination timing plan guideline of this algorithm implies, (1) coordinated phase selection, (2) offset assignment, (3) phase sequence determinations, (4) split assignment, (5) optimum cycle length calculation, (6) Effective green time calculation, (7) phase length calculation, and (8) transition mode definition.

The development of this algorithm starts by presenting a traffic model with its parameters (see Figure 4.2.) to facilitate the understanding of all details given in this section.

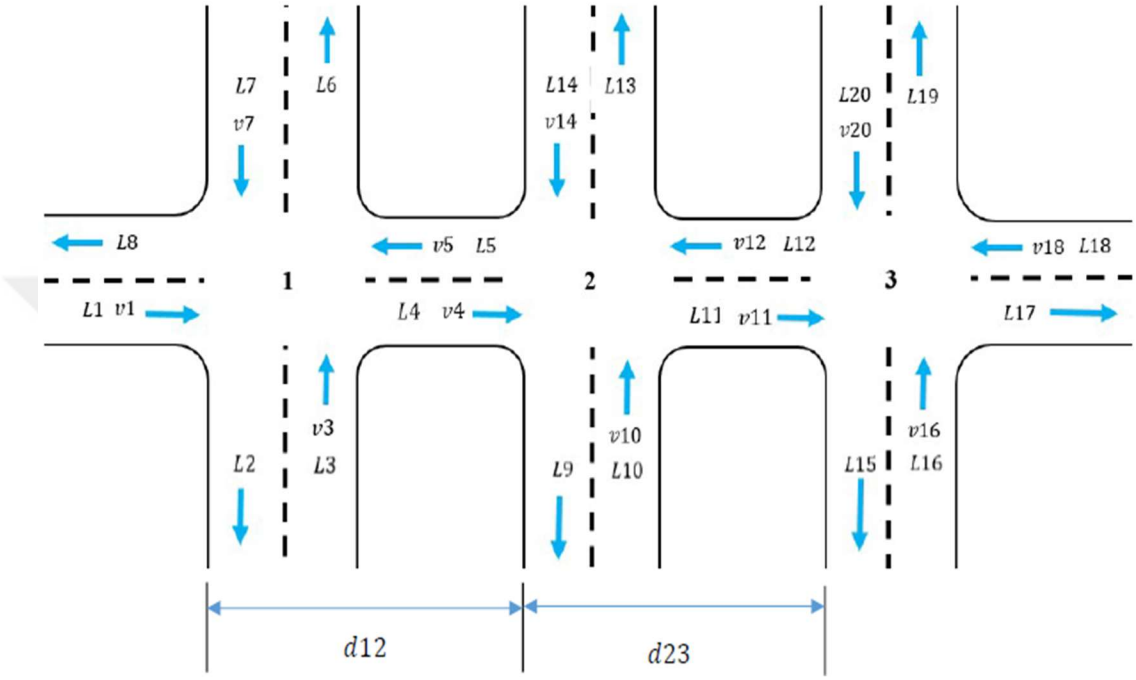

Figure 4.2. Proposed traffic model

Where:

- *d12* is a respective distance between intersection 1 to intersection 2,
- *d23* is a respective distance between intersection 2 to intersection 3,
- *L1*, *L4*, *L5, L8, L11, L12*, *L17, L18* = Major street lines,
- *L2*, *L3*, *L7, L9, L10, L13*, *L14, L15*, *L16*, *L19, L20* = Minor street lines, and
- *v1, v3, v5, v7, v10, v12, v14, v16, v18, v20* = Observed traffic volume (vehicle number of road lines).

4.2.1. Coordinated phase selection and offset assignment

For ensuring the relationship between intersections, the phase 1 is assigned to the movement A or to the movement C is selected to be the coordinated phase. This phase provides the vehicle movements through the major road lines (see Figure 4.4.). In addition, the offset point or offset time duration computed at each controller should be added to this coordinated phase in order to provide an alignment of the coordinated phases between intersections (NCHRP, 2015). Offset time should be computed by using the following equation:

$$Offset = \frac{di}{Sp} \tag{4.1}$$

Where:

- di = Distance between intersections, and

- Sp = Limited speed of the road line.

Figure 4.3. depicts a proposed example of offset times in which time arrangements establish a green wave between the intersections.

Figure 4.3. shows the offset of the intersection 1 set to 15 seconds, of the intersection 2 to 20 seconds and of the intersection 3 to 25 seconds. For a cycle length of 100 seconds, the intersection 1 coordinated phase starts its yellow at 15 seconds and 100 seconds. The intersection 2 coordinated phase starts its yellow at 20 seconds and 120 seconds, therefore 20 seconds after the intersection 1, and the intersection 3 coordinated phase starts its yellow at 25 seconds and 125 seconds, therefore 25 minutes after the intersection 1.

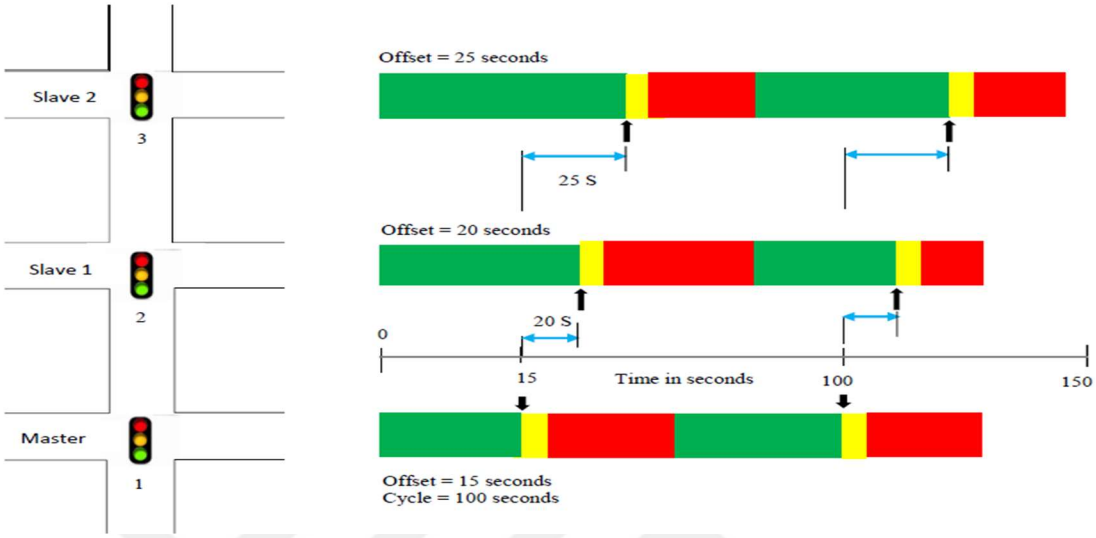

Figure 4. 3. Coordinated Phase with Offsets

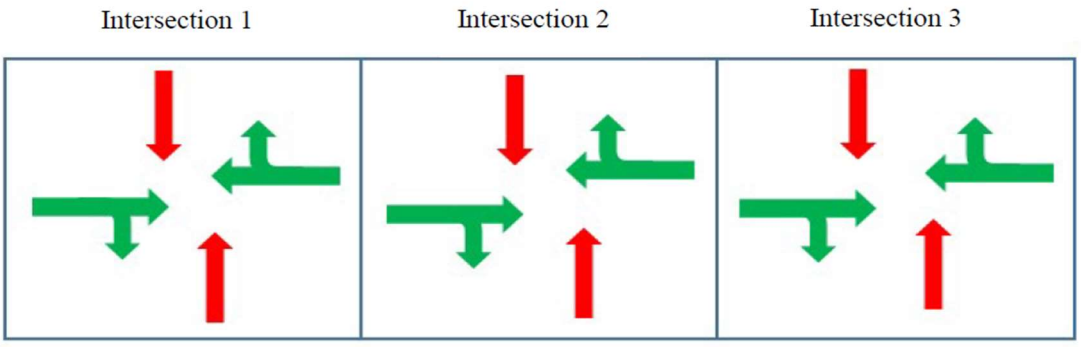

Figure 4.4. Coordinated phase movements

4.2.2. Phase sequence determinations

The proposed phase sequences contain four movements as shown in Figure 4.5., and their arrangement is based on the procedure of comparing the road line traffic volume (see Table 4.1.).

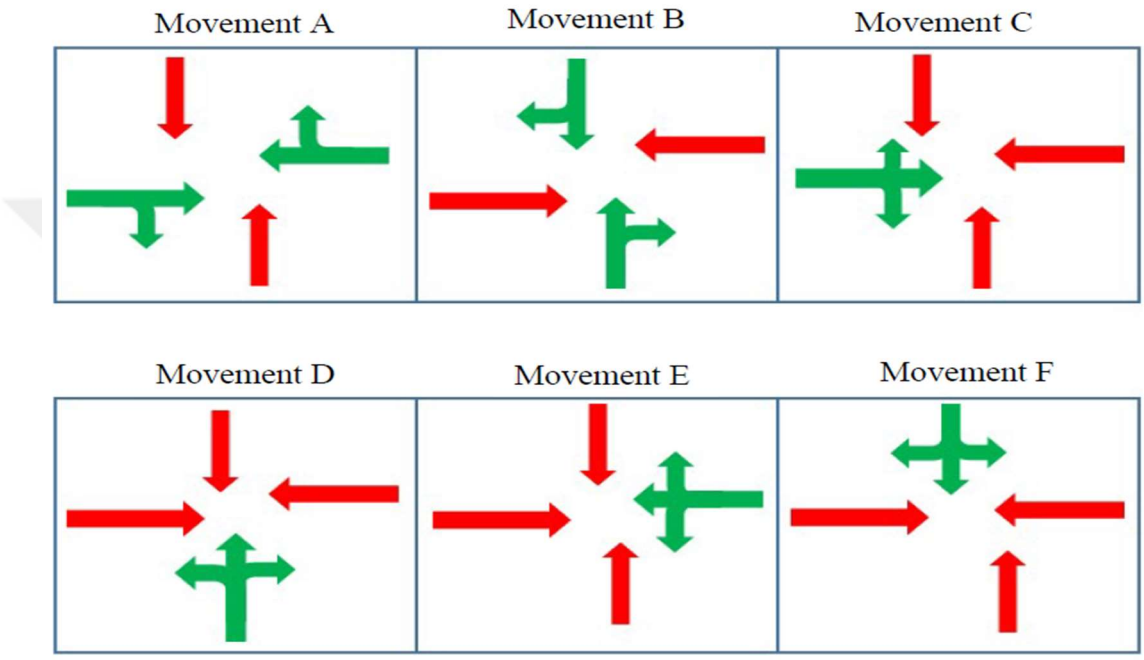

Figure 4.5. Phase movements

During this procedure, the controllers compare the traffic volume located on intersection lines, select the line with the highest traffic volume, which will be served after the coordinated phase (phase 1 + movement A). Then, they select the line with the lowest traffic volume, which will be served by the last phase (phase 4).

The procedure of comparing which should include fifty-four (64) comparison possibilities, and eleven of them are presented in Table 4.1. using an example for intersection 1.

Table 4.1. Traffic volume comparing and phase sequences

Vehicle Number Comparison	Phase 1	Phase 2	Phase 3	Phase 4
If v1 > v3 v3 > v5 v5 > v7	Movement C	Movement D	Movement E	Movement F
If v3 > v1 v1 > v5 v5 > v7	Movement A	Movement D	Movement E	Movement F
If v5 > v1 v1 > v3 v3 > v7	Movement A	Movement E	Movement D	Movement F
If v7 > v1 v1 > v3 v3 > v5	Movement A	Movement F	Movement D	Movement E
If v1 = v3 v3 = v5 v5 = v7	Movement A	Movement B	Movement A	Movement B
If v1 = v3 v3 > v5 v5 > v7	Movement C	Movement D	Movement E	Movement F
If v1 = v5 v5 > v3 v3 > v7	Movement A	Movement B	Movement A	Movement B
If v1 = v7 v7 > v3 v3 > v5	Movement C	Movement F	Movement D	Movement F
If v1 > v3 v3 > v5 v5 = v7	Movement A	Movement B	Movement A	Movement B
If v1 > v5 v5 > v3 v3 = v7	Movement A	Movement B	Movement A	Movement B
If v1 > v7 v7 > v3 v3 = v5	Movement A	Movement B	Movement A	Movement B

Example 1: First, assuming that the road line traffic volumes of the intersection 1 are: *v1* = 645 vph, *v3* = 300 vph, *v5* = 200 vph and *v7* = 100 vph. It is clearly seen that the line 1 has the greatest traffic volume, and the line 7 has the lowest traffic volume. So, according to the first comparison shown in Table 4.1., the phase sequences should be as follows:

- The phase 1 is assigned to the movement C (coordinated phase), which allows the line 1 vehicles to cross the intersection 1,
- The phase 2 is assigned to the movement D, which allows the line 3 vehicles to cross the intersection 1,
- The phase 3 is assigned to the movement E, which allows the line 5 vehicles to cross the intersection 1, and
- The phase 4 is assigned to the movement F, which allows the line 7 vehicles to cross the intersection 1.

Then, assuming that the road line traffic volumes of this intersection 1 are: $v1$ = 270 vph, $v3$ = 240 vph, $v5$ = 548 vph and $v7$ = 150 vph. It is clearly seen that the line 5 has the greatest traffic volume and the line 7 has the lowest traffic volume. So, according to the third comparison shown in Table 4.1., the phase sequences should be as follows:

- The phase 1 is assigned to the movement A (coordinated phase), which allows the line 1 vehicles to cross the intersection 1,
- The phase 2 is assigned to the movement E, which allows the line 5 vehicles to cross the intersection 1,
- The phase 3 is assigned to the movement D, which allows the line 3 vehicles to cross the intersection 1, and
- The phase 4 is assigned to the movement F, which allows line 7 vehicles to cross the intersection 1.

Finally, assuming that the road line traffic volume of this intersection are: $v1$ = 350 vph, $v3$ = 270 vph, $v5$ = 190 vph and $v7$ = 490 vph. It is clearly seen that the line 7 has the greatest traffic volume and the line 5 has the lowest traffic volume. So, according to the forth comparison shown in Table 4.1., the phase sequences should be as follows:

- The phase 1 is assigned to the movement A (coordinated phase), which allows the line 1 vehicles to cross the intersection 1,
- The phase 2 is assigned to the movement F, which allows the line 7 vehicles to cross the intersection 1,

- The phase 3 is assigned to the movement D, which allows the line 3 vehicles to cross the intersection 1, and
- The phase 4 is assigned to the movement E, which allows line 5 vehicles to cross the intersection 1.

4.2.3. Optimum cycle length calculation

An appropriate way to reduce delays is to maintain the cycle lengths as short as possible, since longer cycle lengths increase queue length and capacity at intersections. Computing methods, which include intersection constraints, provide balanced optimum cycle lengths in the range of 60 and 150 seconds (Bonneson, J., Sunkari, S. and Pratt M., 2009).

A critical intersection method that considers intersection constraints such as the lost time and traffic saturation flow rate to compute the effective green time should be used to determine an optimal cycle length. Once that this cycle is found, effective phase green time allocated to the coordinated phases (major movements), and non-coordinated phases (minor movements) should be easily computed. Thus, the formula (2.6) presented in Chapter 2 will be applied to compute this optimal cycle length, which implies:

- Calculation of line flow ratios,
- Selection of critical line flow ratio,
- Calculation of the sum of intersection critical traffic volume, and
- Calculation of total lost time per a cycle length.

Robert, L., Gordon, P.E. and Warren-Tighe, P.E. (2005) proposed a formula to calculate the sum of flow ratios for all critical lanes.

$$Y = \sum_{i=1}^{n} \left(\frac{vi}{Si} \right) \tag{4.2}$$

Where:

- $\left(\frac{vi}{Si} \right)$ = Line flow ratio,

- Y = sum of flow ratios for all critical lanes in phase i.
- vi = Observed traffic volume for phase i (vph),
- Si = Saturation flow rate (vph), and
- n = Number of phase.

They also proposed to calculated lost time per phase that is the time in which an intersection is not efficiently used by a movement. This often occurs at the beginning of a phase. So, this formula is described as follows:

$$tL = l1 + l2 \tag{4.3}$$

Where:

- tL = Lost time in a phase i (sec),
- $l1$ = Start-up lost time, which is the lost time by the vehicle at the beginning of a phase i (sec), and
- $l2$ = Change and clearance interval, which represent the lost time that occurs between signal phases (from phase i to phase i+1, sec).

Thus, the total lost time per cycle can be calculated as follows:

$$L = \sum_{i=1}^{n} tL \tag{4.4}$$

Example 2: Assuming that:

- An ideal saturation flow rate is equal to 1900 vph as described by (Hamad, K. and Abuhamda, H. 2015; Leong, L.V., Wan, H.W.I. and Mohd-Sadullah, A.F. 2005).
- Lost time per phase = 2 seconds like used in the Highway Capacity Manual (TRB, 2016).

For the intersection 1, the road line traffic volumes are as follows, $v1$= 645 vph, $v3$ = 300 vph, $v5$ = 200 vph and $v7$ = 100 vph. So, line flow ratios are:

$$\left(\frac{vi}{Si}\right)=\frac{645}{1900}=0.3394;\frac{300}{1900}=0.1579;\frac{200}{1900}=0.1053;\frac{100}{1900}=0.0526$$

Thus, the highest value 0.3394 is the critical flow ratio for the intersection 1.

For the intersection 2, the road line traffic volumes are as follows, *v4* = 270 vph, *v10* = 240 vph, *v12* = 548 vph and *v14* = 150 vph. So, line flow ratios are:

$$\left(\frac{vi}{Si}\right)=\frac{270}{1900}=0.1421;\frac{240}{1900}=0.1263;\frac{548}{1900}=0.2884;\frac{150}{1900}=0.0789$$

Thus, the highest value 0.2884 is the critical flow ratio for the intersection 2.

For the intersection 3, the road line traffic volumes are as follows, *v11* = 350 vph, *v16* = 270 vph, *v18* = 190 vph and *v20* = 490 vph. So, line flow ratios are:

$$\left(\frac{vi}{Si}\right)=\frac{350}{1900}=0.1842;\frac{270}{1900}=0.1421;\frac{190}{1900}=0.1;\frac{490}{1900}=0.2579$$

Thus, the highest value 0.2579 is the critical flow ratio for the intersection 3.

Therefore sum of critical flow ratios is:

$$Y=\sum_{i=1}^{n}\left(\frac{vi}{Si}\right)\rightarrow Y=0.3394+0.2884+0.2579=0.8857$$

As seen in Table 4.1., there are four (4) phases per cycle, and therefore the total lost time per cycle can be calculated as follow:

$$L=4(2)=8 \text{ seconds}$$

Finally the optimal cycle is:

$$C_{opt} = \frac{1.5L + 5}{1 - Y} \rightarrow C_{Opt} = \frac{1.5(8) + 5}{1 - 0.8857} = 148.73 \text{ seconds}$$

This computed cycle length duration is in the required range of 60 to 150 seconds as mentioned Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. (2009), The optimal cycle length formula should allow to compute sufficient phase green time durations as mentioned in the Ontario Traffic Manual (OTM, 2001). It is added that the typical maximal computed cycle length can reach 180 seconds, which should allow to obtain typical phase green time durations within the range of 10 to 40 seconds for minor approaches, and within the range of 30 to 120 seconds for major approaches.

Thus, the computed optimal cycle length should be in the range of 60 to 180 seconds. If its value is greater than 180 seconds, the value of 180 must be used in order to compute sufficient phase green time durations remaining within the range of 10 to 120 seconds.

As declared by the Timing Signal Manuals NCHRP. (2015) and MnDOT. (2017), the coordination should be used for close intersections with high traffic volume, which provide acceptable intersection flow ratios that positively affect the optimal cycle length value. In addition, this contributes to the reduction of delays and travel times.

In the case of very high traffic volume, the computed optimal cycle will probably be greater than 60 or 180 seconds. Thus, it is preferable to set limit values for the sum of critical line flow ratios.

These limit values must be like this:

$$Y = \left(\frac{vi}{Si} \right) < 1$$

Since if $Y = \left(\frac{vi}{Si} \right) >= 1$, the optimal cycle length will be either equal to an infinite value or to negative value, which will negatively affect the effective green time.

Example 3: Assuming that $Y = 1$ and $Y = 2$, the optimal cycle length should be like this:

$$C_{opt} = \frac{1.5(8)+5}{1-1} = \frac{17}{0} = \infty \text{ (infinite value)}$$

$$C_{opt} = \frac{1.5(8)+5}{1-2} = \frac{17}{-1} = -17 \text{ (negative value)}$$

4.2.4. Effective green time and phase length calculation

Once the optimal cycle length is calculated, the effective green time that will be allocated to each phase should be calculated by using the formulas (2.10) to (2.13).

Example 4: Assuming a yellow time of 4 seconds per phase, which is described as yellow time for the critical conditions by Summala, H. (2000). Also, and the total lost time and the optimal cycle length are kept the same as found in example 2.

The net green time or split green time is computed as follows:

$$G_t = C_{Opt} - \sum_{i=1}^{n} yell_n - L \rightarrow G_t = 148.73 - 4(4) - 8 = 124.73 \text{ seconds}$$

As seen in example 1, the phases are allocated to the movements according to the vehicle number comparison. Here, these allocations will permit to compute the sum of critical volume of intersection and the green time of each phase.

With the intersection 1, *v1*= 645 vph, *v3* = 300 vph, *v5* = 200 vph, and *v7* = 100 vph. So, the sum of critical volume for this intersection should be computed as follows:

$$V = \sum_{i=1}^{n} vi \rightarrow V = v1 + v3 + v5 + v7 = 645 + 300 + 200 + 100 = 1245 \text{ vph}$$

- The phase 1 implies the movement C, which only leads the line 1 vehicles to across intersection in three directions, left, right, and straight (see Figure 4.5.). The green time for this phase is computed as follows:

$$G_i = G_t\left(\frac{vi}{V}\right) \rightarrow G_1 = G_t\left(\frac{v1}{V}\right) = 124.73\left(\frac{645}{1245}\right) = 64.62 \text{ seconds}$$

$$\text{Phase duration1} = G_i + y_n + tL = 64.62 + 4 + 2 = 70.62 \text{ seconds}$$

- The phase 2 implies the movement D, which only leads the line 3 vehicles to across in three directions, left, right, and straight (see Figure 4.5.). The green time for this phase is computed as follows:

$$G_3 = G_t\left(\frac{v3}{V}\right) = 124.73\left(\frac{300}{1245}\right) = 30.06 \text{ seconds}$$

$$\text{Phase duration2} = G_i + y_n + tL = 30.06 + 4 + 2 = 36.06 \text{ seconds}$$

- The phase 3 implies the movement E, which only leads the line 5 vehicles to across in three directions, left, right, and straight (see Figure 4.5.). The green time for this phase is computed as follows:

$$G_5 = G_t\left(\frac{v5}{V}\right) = 124.73\left(\frac{200}{1245}\right) = 20.04 \text{ seconds}$$

$$\text{Phase duration3} = G_i + y_n + tL = 20.04 + 4 + 2 = 26.04 \text{ seconds}$$

- The phase 4 implies the movement F, which only leads the line 7 vehicles to across in three directions, left, right, and straight (see Figure 4.5.). The green time for this phase is computed as follows:

$$G_7 = G_t\left(\frac{v7}{V}\right) = 124.73\left(\frac{100}{1245}\right) = 10.02 \text{ seconds}$$

$$\text{Phase duration2} = G_7 + y_n + tL = 10.02 + 4 + 2 = 16.02 \text{ seconds}$$

$$\text{Sum of phase duration} = \sum_{i=1}^{n} \text{Phase duration}$$

$$\text{Sum of phase duration} = 70.62 + 36.06 + 26.04 + 16.02 = 148.74 \text{ seconds}$$

This sum is equal to the computed optimal cycle length as said in the Timing Signal Manuals (MnDOT, 2017). In addition, the greatest green time (64.62 seconds) is allocated to the line 1, which has the greatest traffic volume (645 vph).

4.2.5. Transition mode définition

According to NCHRP. (2015) and Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. (2011), transition logic is a process used for the goal of timing plan changes. This permits controllers to update traffic data at a predetermined time (e.g. every hour, a.m., off-peak, or p.m.), and to compute new parameters such as cycle length and phase green times.

4.3. Proposed Algorithm Description

The tasks included in the algorithm presented below in Figure 4.6. are going to be performed as follows. Firstly, an optimal cycle time and a Transition Value duration (TV) must be assigned (i.e., $C_{opt} = 75$ seconds and $TV = 5$ minutes). These values are sent to all three controllers.

Secondly, the master controller first processes traffic data (traffic volumes) measured at intersections. Then, all controllers (master and slaves) compute the flow ratio and sum of flow ratio using the formula (4.2.). Finally, they compute the sum of critical traffic volume

(V), net green time (G_t) and green times (G_i) using the formulas (4.3), (4.4), and these from (2.10) to (2.12).

Thirdly, each controller performs the comparison road line traffic volumes, which allow to determine the most and lowest crowed road lines. It completes the cycle duration by first allocating the green time durations to the coordinated phase, then to the most crowded road line, continuing to the moderately crowded road line, and finally to the lowest crowded road line.

Finally, at the end of each cycle, it is checked whether the transition value (TV) is equal to zero. When this condition is filled, the traffic data will be updated again, the new V, G_t, and G_i will be computed (updated). Furthermore, at this stage, the optimal cycle length C_{opt} assigned to the intersections is calculated using the formula (2.6), and assigned as the new cycle time duration.

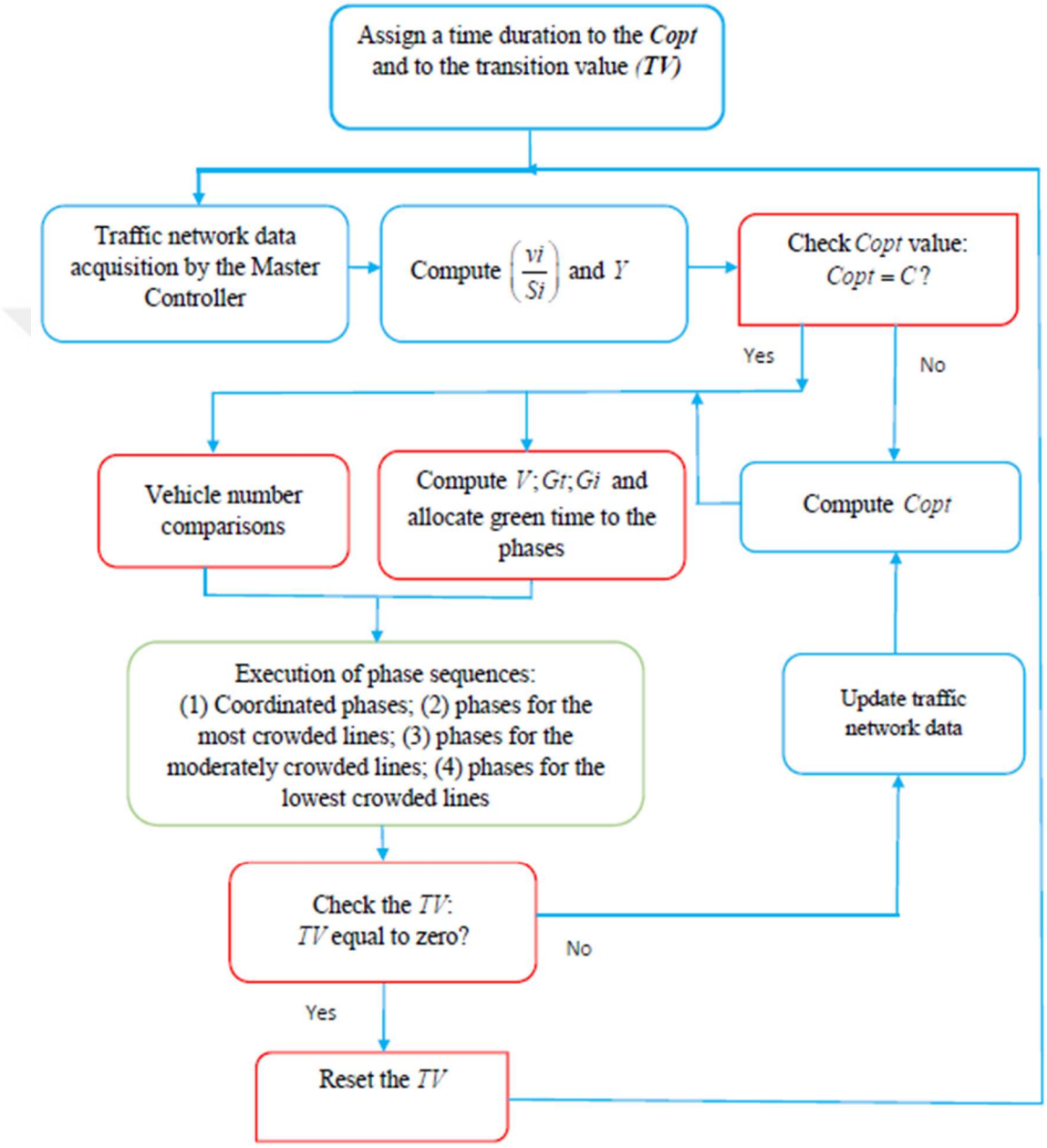

Figure 4.6. Proposed algorithm

4.4. Prototype Implementation

Accordind to Oladimeji, T. and Oshevire, P. (2014), the physical implementation of the technical project is very useful since it allows to perform real tests for being sure of its proper functioning in real time. Beyond this importance, the prototype implementation was motivated by the fact that it is not always easy to obtain an agreement from the municipal authorities to test an algorithm on the real traffic network.

4.4.1. Hardware and software

As mentioned before, the traffic model carried out in the prototype consists of three traffic intersections. Each of them contains four roads with two traffic lines, controllers, traffic light modules and potentiometers. The complete circuit diagram is shown below (Figure 4.7.).

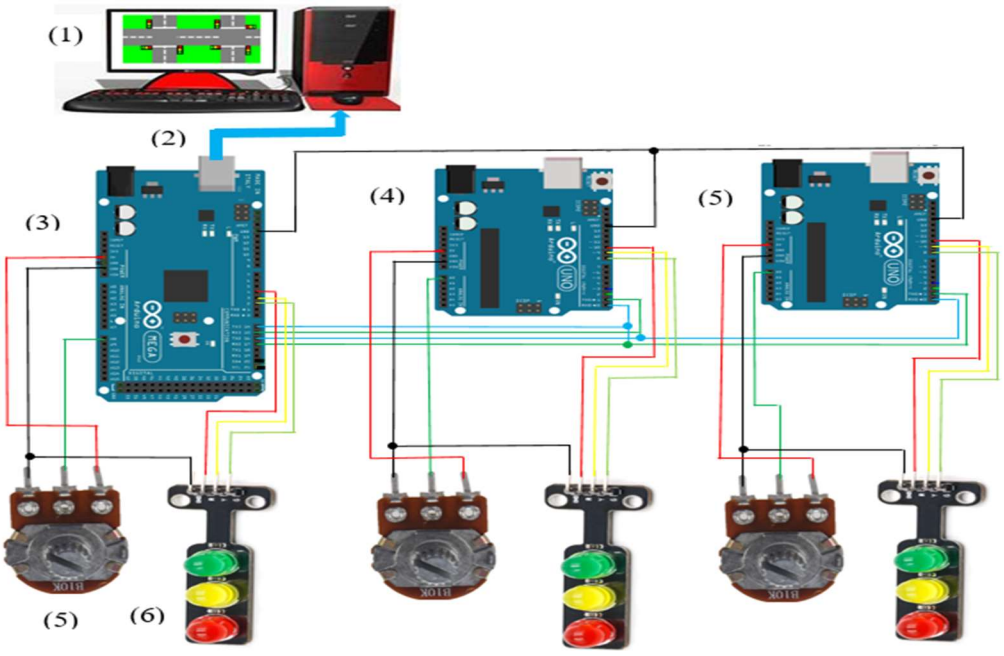

Figure 4.7. Connection Circuit Diagram

Table 4.3. depicts the list of component names, software and hardware used for the implementation of the prototype. This implementation includes, (i) hardware and software presentation, (ii) description of operation mode, (iii) assembly of components, and (iv) communication process between the controllers.

Table 4.2. Software and Hardware

Hardware			**Software**
No	**Components**	**Qty**	Arduino IDE
1	Laptop (Computer)	1	Arduino Virtualwire library
2	Platinum-Power USB PC Cable Cord	3	Arduino Wire library
3	Arduino Mega (Master)	1	
4	Arduino Uno (Slave)	2	
5	Potentiometer	12	
6	Traffic Light Module	12	
7	Jumper Cable	156	
8	Plastic Traffic Platform	2	
9	Electronic Breadboard	3	
10	Multiple port USB adapter	1	
11	Pasteboard	3	

4.4.2. Component connections

Table 4.4. shows the type of pins, analog and digital as well as their location number used to connect all electronic components of the system.

The potentiometers used have three pins (see Figure 4.7.), the left one is connected to the 5v pin of controller, the right to the GND pin, and the middle to analog pins, in particular from the pin A0 to A3. The traffic lights modules have four pins (see Figure 4.7.) named as GND for ground, R for red light, Y for yellow light and G for green light. The pin GND is connected to the pin GND of controllers, the pins R, Y and G to digital pins, particularly from the pin 2 to 13.

Table 4.3. Component pins connection

Intersections	Line Potentiometers	Controller pins used (Analog)	Line Lights	Controller pins used (Digital)
Intersection1 (Master)	Line1_Potentiometer	A0	Green; Yellow; Red	2; 3; 4
	Line3_Potentiometer	A1	Green; Yellow; Red	5; 6; 7
	Line5_Potentiometer	A2	Green; Yellow; Red	8; 9; 10
	Line7_Potentiometer	A3	Green; Yellow; Red	11; 12; 13
Intersection2 (Slave1)	Line4_Potentiometer	A0	Green; Yellow; Red	2; 3; 4
	Line10_Potentiometer	A1	Green; Yellow; Red	5; 6; 7
	Line12_Potentiometer	A2	Green; Yellow; Red	8; 9; 10
	Line14_Potentiometer	A3	Green; Yellow; Red	11; 12; 13
Intersection3 (Slave2)	Line11_Potentiometer	A0	Green; Yellow; Red	2; 3; 4
	Line16_Potentiometer	A1	Green; Yellow; Red	5; 6; 7
	Line18_Potentiometer	A2	Green; Yellow; Red	8; 9; 10
	Line20_Potentiometer	A3	Green; Yellow; Red	11; 12; 13

For the communication between the Arduino boards, the master's pin TX (14) is connected to the Slave 1's pin RX, and its pin RX (15) is connected to the Slave 1's pin TX. Also, its pin TX (16) is connected to the slave 2's pin RX, and its pin RX (17) is connected to the Slave 2's pin TX. In addition, The slave 1's pin TX is connected to the slave 2's pin RX, its pin RX is connected to the Slave 2's pin TX, and the controllers pins GND are connected.

4.4.3. Mode of operation

The concept of operating mode of this system is based on the vehicle number detected on the road lines. Once the potentiometer resistance values are varying, the controllers read those values as an analog value (Shah, S. and Shah, U. 2016; Baichtal, J. 2014). Those values indicate the traffic volume (vehicle number) detected on traffic lines. Then, Slave controllers send those values to each other, and to the Master controller, which compute cycle length duration, and sends that to the slave controllers. Finally, all controllers compute the phase green time and turn on the green lights as well as red lights according

to the traffic volume comparison as mentioned early in the algorithm definition. Furthermore, the transmitted data between controllers are displayed thanks to the computer (laptop) screen.

4.4.4. Prototype construction

The prototype was built in the shape of a square box having a hole in which electronic components such as controllers and jumper cables are placed. Its construction begins by pasting the three pasteboards each other, and by attaching them bellow from the top plastic platform. Then, the traffic road and lines were designed atop this plastic platform. Also, the potentiometers and the traffic light modules were mounted on this platform. Finally, the controllers were wired to the potentiometers and the traffic light modules, and placed between the top and bottom plastic platform as shown below (Figure 4.8.). All the prototype components have been pasted by using sticky tapes rather than glue.

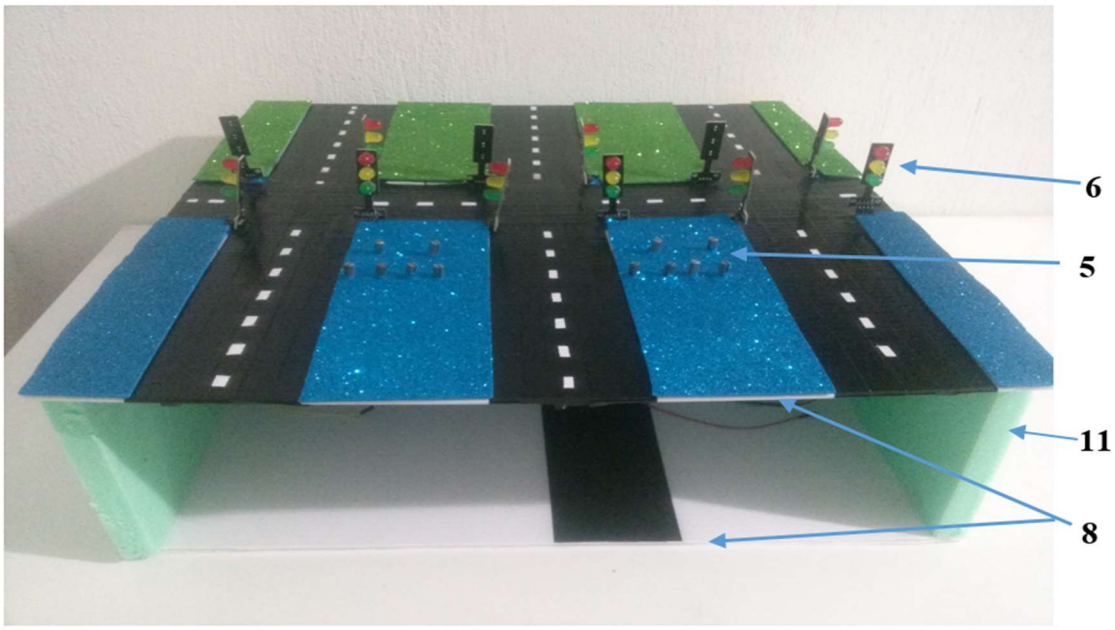

Figure 4. 8. Prototype Implemented

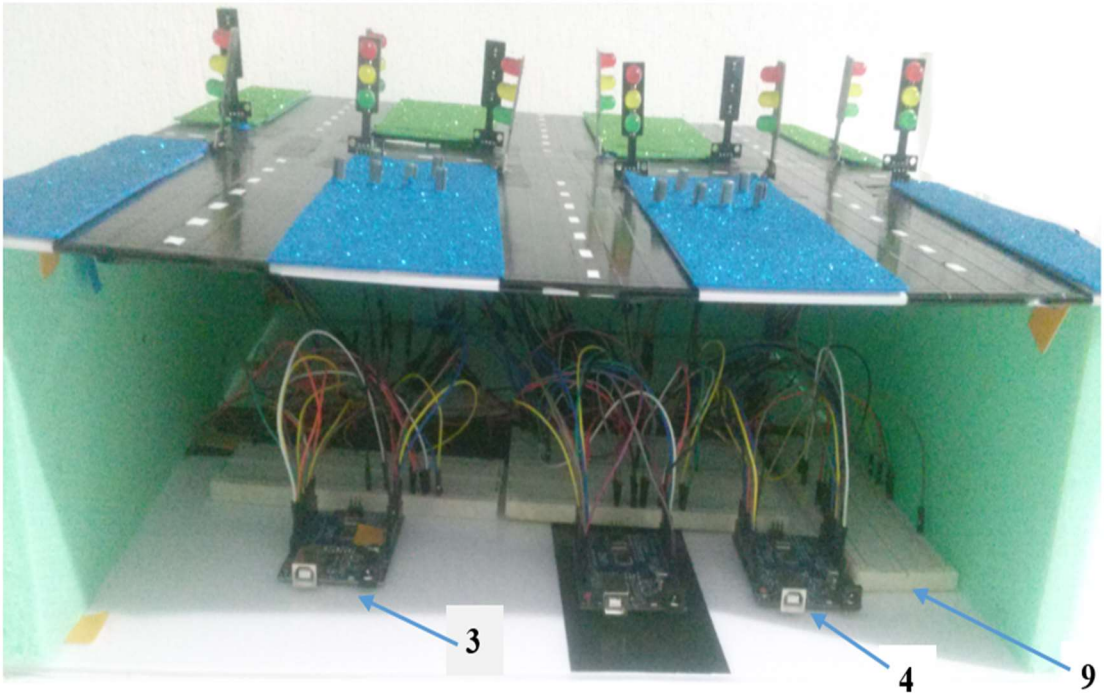

Figure 4.9. Controllers view on the Prototype

4.4.5. Communication between controllers

According to Gordon, R.L. and Tighe, W. (2005), a communication between controllers is necessary since it allows them (i) to distribute a common cycle length, upload and download timing plans and traffic data, (ii) to execute adaptive and coordinated algorithms, and (iii) to supervise the transmission of data required for the emergency vehicle control, traffic monitoring and incident detection components. Thus, our proposed communication between controllers targets the above functions (i) and (ii) in order to respect the coordinated and adaptive control systems requirements.

Among the communication technics between the Arduino Boards described by (Shah, S. and Shah, U. 2016; Baichtal, J. 2014), we proposed to use the wired serial communication, which will be implemented in the prototype. This communication technic was carried out

by connecting Arduinos to each other with jumper cables as stated earlier. The real connection of the entire system is shown below (Figure 4.10.).

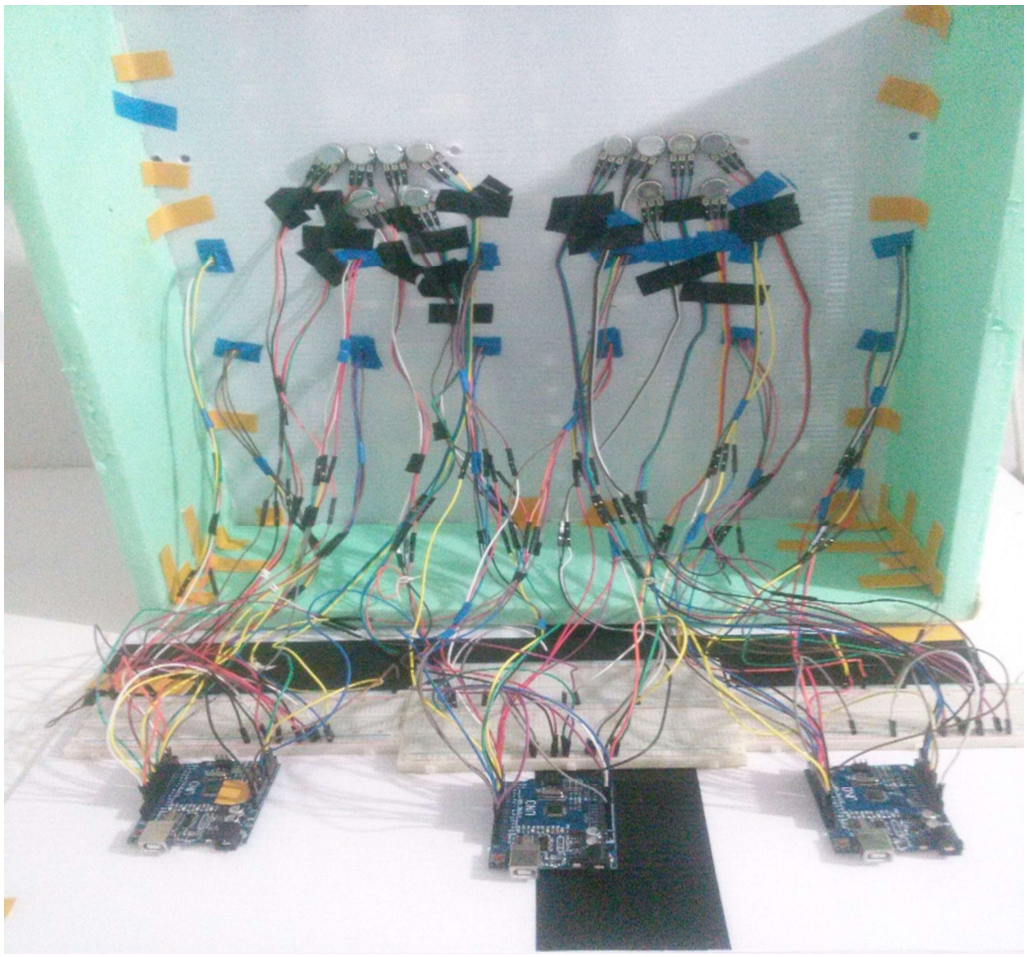

Figure 4.10. Entire system connection

5. EXPERIMENTAL TESTS AND RESULTS

This chapter presents the different steps undertaken during the experimental simulation and tests as well as the test results. The first step was to program the proposed algorithm within the Arduino IDE software, and the second was to perform the tests.

5.1. Algorithm Coding

The coding started by defining the variables for all intersections. Figure 5.1. shows how this was carried out for the intersection 1. First, the variables Line1_Potentiometer, Line3_Potentiometer, Line5_Potentiometer, and Line7_Potentiometer represent the potentiometers placed on the line 1, line 3, line 5 and line 7, which were connected to the analog pins A0, A1, A2 and A3.

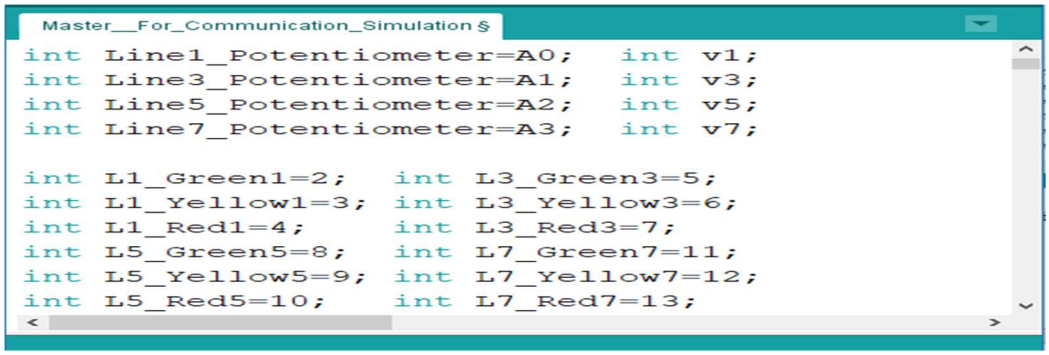

Figure 5.1. Definition of the variables

Then, the variables v1, v3, v5, and v7 represent the variables that storage resistance values coming from the potentiometers, which indicate the traffic volumes (vehicle numbers) on the road lines. Finally, the line light variables are connected to the digital pins, i.e., L1_Green1 = 2 means that the line 1 green light is connected to the digital pin 2.

5.1.1. Coding for data sending and phase green time calculation

As mentioned in the algorithm definition (chapter 4), the program must start with a predefined cycle length and transition value (*TV*). Thus, the cycle length was set to 75 seconds and *TV* to 5 minutes. Also, the lost time (*tL*) per phase was set to 2 seconds, the yellow time (*Yell*) per phase was also set to 4 seconds. So, Figure 5.2. shows the coding for data sending, and the calculation of the sum of critical line volume (*V*) and yellow time, as well as the net and phase green times (G_t and G_i).

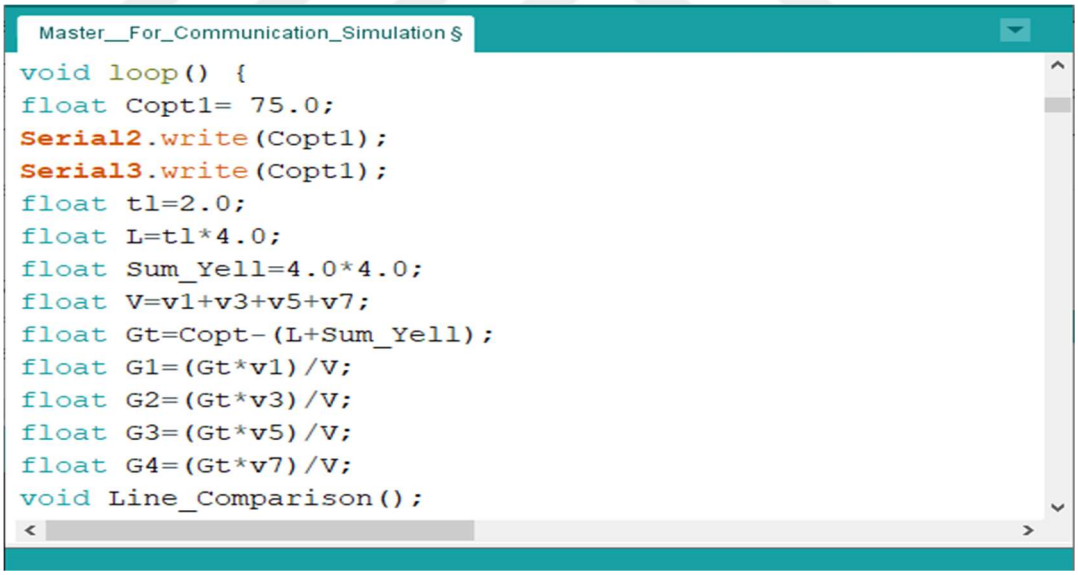

Figure 5.2. Coding for data sending and phase green time calculation

The functions “serial2.write” and “serial3.write” were used to send the cycle length from the master controller to the slave controllers. In addition, the simple function “serial.write” was used to send vehicle numbers from the slave controllers to the master controller. The

function “Line_comparison” was defined to compare vehicle numbers and to execute the phase sequences. The content of this function is illustrated in Appendix A.

5.1.2. Coding for data receiving

As shown in Figure 5.3., the function “Serial2.readString” first reads vehicle numbers merged in one string data type coming from the Slave 1, and affects this to the variable “TrafficDataFromSlave1”. Then, this data is separated in four string data, and affected to the variables Value11, Value13, Value14, and Value15. Finally, these four data are converted into float data type and assigned to “Slave1_Line4_Vehicle_Number”, “Slave1_Line10_Vehicle_Number” etc.

The string data type was separated with the help of the “indexOf” and “Substring” functions. The conversion of these data were made by using the function “toFloat”. More details about these functions may be found from the Arduino community website.

Master__For_Communication_Simulation §

```
String TrafficDataFomSlave1=Serial2.readString();
String Value11=TrafficDataFomSlave1.substring(3,TrafficDataFomSlave1.indexOf('b'));
String Value12=TrafficDataFomSlave1.substring(TrafficDataFomSlave1.indexOf('b')+3,TrafficDataFomSlave1.indexOf('c'));
String Value13=TrafficDataFomSlave1.substring(TrafficDataFomSlave1.indexOf('c')+3,TrafficDataFomSlave1.indexOf('d'));
String Value14=TrafficDataFomSlave1.substring(TrafficDataFomSlave1.indexOf('d')+3),TrafficDataFomSlave1.indexOf('e');
float Slave1_Line4_Vehicle_Number=Value1.toFloat();
float Slave1_Line10_Vehicle_Number=Value2.toFloat();
float Slave1_Line12_Vehicle_Number=Value3.toFloat();
float Slave1_Line14_Vehicle_Number=Value4.toFloat();
```

Figure 5.3. Coding for data receiving

The function “Serial3.readString” was used to read data coming for slave 2 controller. Moreover, the simple “Serial.read” function was used to read cycle length coming from the master controller to the slave controllers.

5.1.3. Coding for cycle length calculation and data display

In order to compute a new cycle every 5 five minutes (50000 milliseconds), the “millis” function was used. A counter up was established with the help of an if loop which includes the unsigned long variables such as “ intervalTime” set to 50000, “previousTime1” set to zero, and “currentTime” set equal to the “millis” function (see Figure 5.4.).

For computing the cycle length, the saturation flow rate was set to 1900 vph, which allowed to compute the line flow ratios.

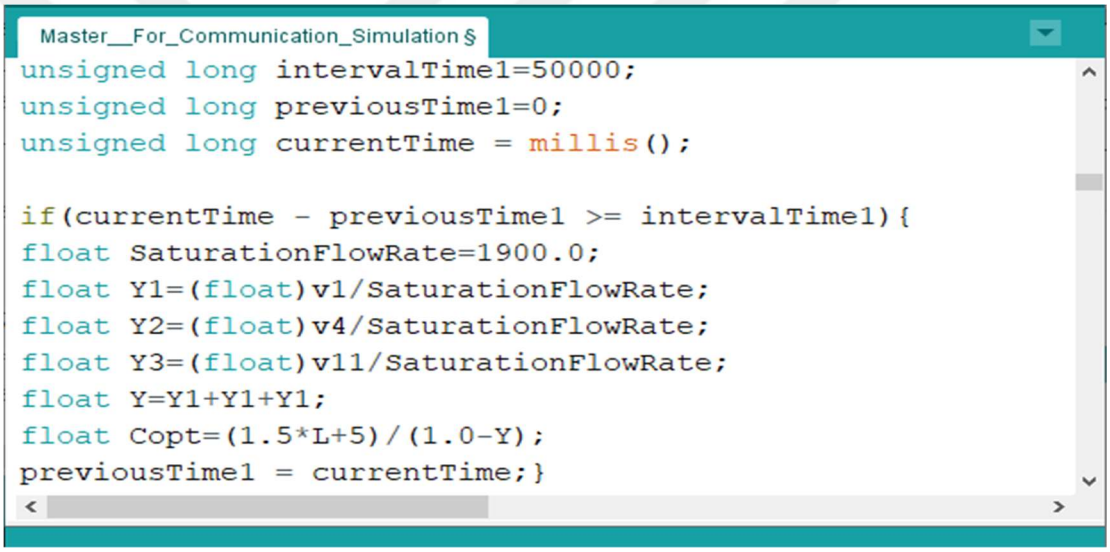

Figure 5.4. Coding for cycle length calculation

Figure 5.5. below shows, the coding for displaying sent and received traffic data. The functions such as “Serial.print” and “Serial.println” which print data to the IDE serial port were used. More explanations about these functions were given by Shah, S. and Shah, U. (2016) and Baichtal, J. (2014).

The sentences written in parentheses from “Serial.print” are displayed first as shown in Figure 5.7., and the variables set in parentheses from Serial.println include the value displayed after (front of comments). Delay time command was used to allow the display of data every 500 milliseconds.

Slave1_For_Communication_Simulation

```
Serial.print(" String Vehicle Number Sent to the Master is : ");
Serial.println(TrafficDataSentToMaster);
Serial.println(" ");
Serial.print(" Cycle length  received from the Master in seconds is: ");
Serial.println(Copt);
Serial.println(" ");
Serial.print(" The Phase Green Times are: ");
Serial.println(" ");
Serial.print(" Phase1 Green Time in seconds is: ");
Serial.println(G1);
Serial.print(" Phase2 Green Time in seconds is: ");
Serial.println(G2);
Serial.print(" Phase3 Green Time in seconds is: ");
Serial.println(G3);
Serial.print(" Phase4 Green Time in seconds is: ");
Serial.println(G4);
Serial.println(" ");
delay(500); }
```

Figure 5.5. Code for displaying phase green times and sent received data

5.2. Test Results

As mentioned before, the implementation of a prototype was performed in order to evaluate the reliability and feasibility of the proposed algorithm, as well as its capabilities to respond in real time demands of traffic. For these ends, the experimental tests was carried out and the results were displayed with the help of, (i) the codes described previously, (ii) the IDE software serial monitor screen, and (iii) to the physical connection of the components like shown below (Figure 5.6.).

This process of testing and simulation was very complex due to the control of the three intersections at the same time, as well as to the large amount of traffic data that should be controlled and monitored during their transmissions. In addition, more attention has been paid to the traffic data transferred in order to avoid data loss, which should negatively affect the experiment test results.

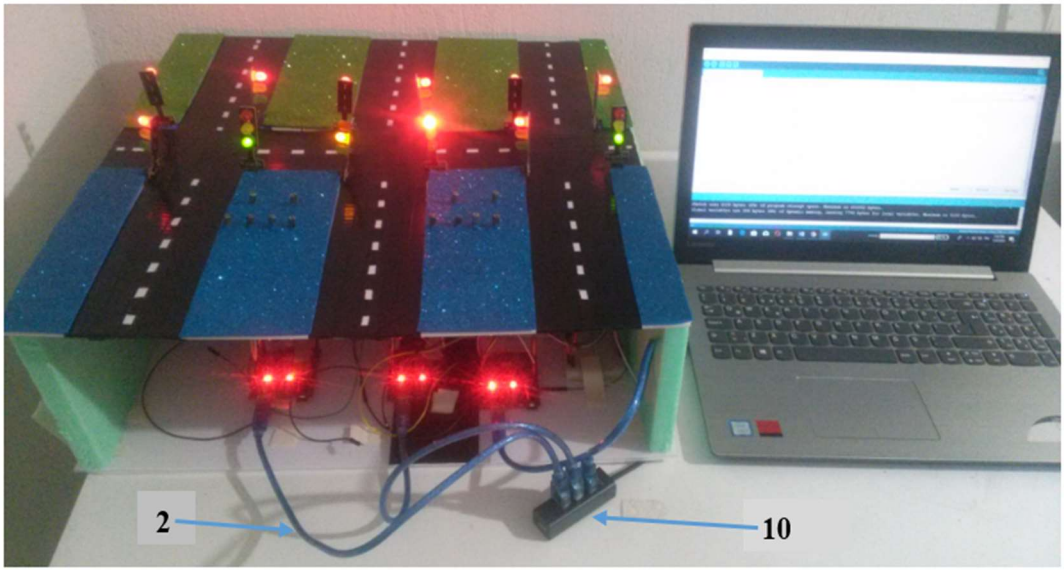

Figure 5.6. Test procedure

The test procedures were performed in two stages. The first stage was to simulate the transmission of traffic data between intersections, which is related to the commutation between the controllers. The second stage implied two cases, whose the first case was to test the prototype with the proposed algorithm (adaptive control). The second case was to test the prototype with a fixed traffic control. The experimental test results obtained from these cases were benchmarked and commented.

5.2.1. Simulation for the communication between the intersections

This simulation involved the following stapes, (i) the slave controllers sent the lines vehicle numbers to the master controller, (ii) the master controller sent the cycle length to the slave controllers, and (iii) the display of all these parameters as shown below (in Figure 5.22.). These stapes were performed several times in order to be sure of the transmission without the loss of data. Also, these tests were done with low traffic volume (vehicle number).

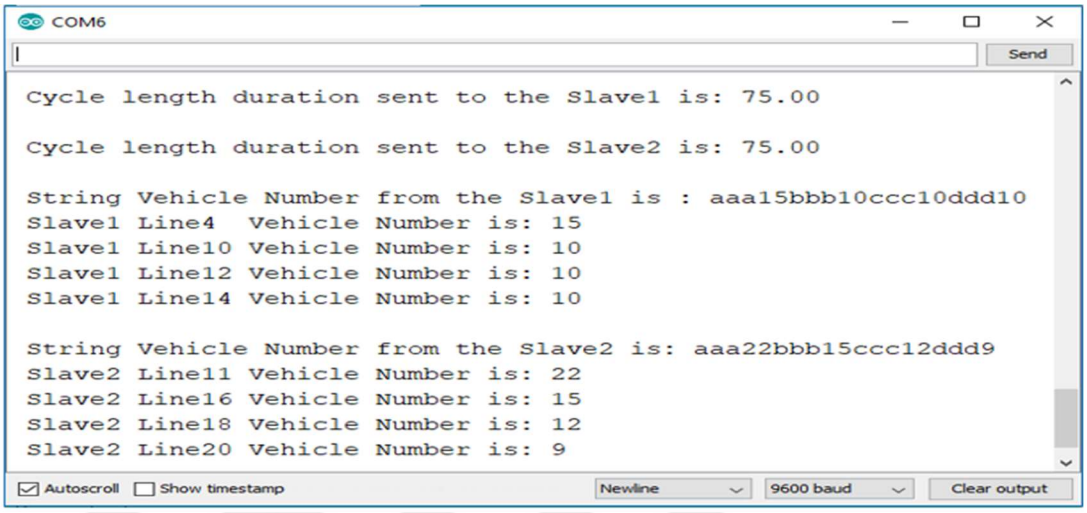

Figure 5. 7. Sent and received data displaying

5.2.2. Prototype test with the proposed algorithm

5.2.2.1. First tests

These tests were performed with high traffic volume. The controllers computed the green time durations using all formulas defined in Chapter 4.

In order to obtain interpretable results in this adaptive control case (first case), the intersections were tested separately according to the following scenarios. First, the road line 1 (L1) traffic volume of the intersection 1 were increasing per 75 vph every five minutes, starting by 300 vph. The traffic volume of demands of other road lines were maintained constant to 300 vph, 200 vph, and 100 vph like it was used for the example 2 and example 3 (Chapter 4).

Then, the road line 4 (L4) traffic volume of the intersection 2 were also increasing per 75 vph every five minutes starting by 300 vph while the traffic volume of other road lines were maintained constant to to 270 vph, 240 vph, and 150 vph.

Finally, the road line 11 (L11) traffic volume of the intersection 3 were also increasing per 75 vph every five minutes, starting by 300 vph while the traffic volume of other road lines were maintained constant to to 350 vph, 270 vph, and 190 vph.

The experimental test results for this first case are presented below in Table 5.1., they show how much the values of green light time durations allocated to the phase 1 related to L1, L4, and L11 rapidly increased (varied) when the road line traffic volumes increased. Moreover, they also show the fast variations of the computed optimal cycle length values during these road line traffic volumes increases.

In the fixed-time control case (second case), the traffic volume of the road line 1 (L1), line 4 (L4), and line 11 (L11) were also increasing per 75 vph, starting by 300 vph every five minutes, while the traffic volume of other road lines were maintained constant to 300 vph, 200 vph, and 100 vph like it was used for the example 2 and example 3 (Chapter 4).

The experimental test results for this first case are also presented in Table 5.1., they show how the values of green light durations allocated to the phase 1 related to L1, L4, and L11 are remained constant as well as the values of the computed cycle length.

Table 5. 1. Test results for high traffic volume (Adaptive control)

Adaptive control					
Intersection 1 L1 Traffic Volume (vp5mins)	**Phase 1** (L1,L2); (L1,L4) (L5,L6); (L5,L8)	**Phase 2** (L7,L2); (L7,L8); (L3,L4); (L3,L6)	**Phase 3** (L1,L2); (L1,L4); (L1,L6)	**Phase 4** (L5,L2); (L5,L6); (L5,L8)	**Computed Optimal Cycle length** (in seconds)
	Green Time (in seconds)	**Green Time (in seconds)**	**Green Time (in seconds)**	**Green Time (in seconds)**	
300	11.17	10.14	7.44	3.72	57.47
375	16.28	13.02	8.68	4.34	66.32
450	23.31	15.54	10.36	5.18	78.39
525	33.53	19.16	12.77	6.37	95.85
600	49.64	24.82	16.55	8.27	123.28
675	78.74	34.99	23.33	11.66	172.72
Intersection 2 L4 Traffic Volume (vp5mins)	**Phase 1** (L4,L9); (L4,L11) (L12,L5); (L12,L13)	**Phase 2** (L14,L5); (L14,L9); (L10,L11); (L10,L13)	**Phase 3** (L4,L9); (L4,L11); (L4,L13)	**Phase 4** (L12,L5); (L12,L9); (L12,L13)	**Computed Optimal Cycle length** (in seconds)
300	14.21	12.78	11.37	7.10	69.46
375	21.31	15.34	16.98	10.61	82.52
450	31.84	19.10	16.73	16.73	102.54
525	48.99	25.20	22.40	13.99	134.58
600	68.57	30.86	27.43	17.14	150
675	87.98	35.19	31.29	19.55	180
Intersection 3 L11 Traffic Volume (vp5mins)	**Phase 1** (L11,L15); (L11,L17); (L18,L12); (L18,L19)	**Phase 2** (L20,L12); (L20,L15); (L16,L17); (L16,L19)	**Phase 3** (L11,L15); (L11,L17); (L11,L19)	**Phase 4** (L18,L12); (L18,L15); (L18,L19)	**Computed Optimal Cycle length** (in seconds)
300	14.96	17.46	13.47	9.48	79.36
375	23.19	21.64	16.69	11.75	97.29
450	36.31	28.24	21.79	15.33	125.68
525	60.34	40.24	31.04	21.84	177.47
600	74.04	41.19	31.64	22.26	180
675	79.05	43.19	33.22	23.45	180

Table 5. 2. Test results for high traffic volume (Fixed-time control)

Fixed Time Control					
Intersection 1 L1 Traffic Volume (vp5mins)	**Phase 1** (L1,L2); (L1,L4) (L5,L6); (L5,L8)	**Phase 2** (L7,L2); (L7,L8) (L3,L4); (L3,L6)	**Phase 3** (L1,L2); (L1,L4) (L1,L6)	**Phase 4** (L5,L2); (L5,L6) (L5,L8)	**Cycle length (in seconds)**
300	29.65	29.65	29.65	29.65	29.65
375	29.65	29.65	29.65	29.65	89.23
450	29.65	29.65	29.65	29.65	89.23
525	29.65	29.65	29.65	29.65	89.23
600	29.65	29.65	29.65	29.65	89.23
675	29.65	29.65	29.65	29.65	89.23
750	29.65	29.65	29.65	29.65	89.23

5.2.2.2. Comparison of the first test results

Figure 5.8. depicts the comparison of the adaptive control and fixed-time control obtained in Table 5.1. and Table 5.2. In this figure, the x-axis of the graph represents the traffic volume (vehicle number) values, and the y-axis represents the phase green time values. It is clearly seen that for adaptive control, the phase green times (blue, orange and gray curves) rapidly increase when the traffic volumes (vehicle numbers) increase, while other phase green times (yellow curve) slowly increase. Also, the increase of the computed cycle length durations (purple curve) is observed. For fixed-time control, the phase green times (red curve) as well as the cycle length durations (green curve) remain constant while the vehicle numbers increase.

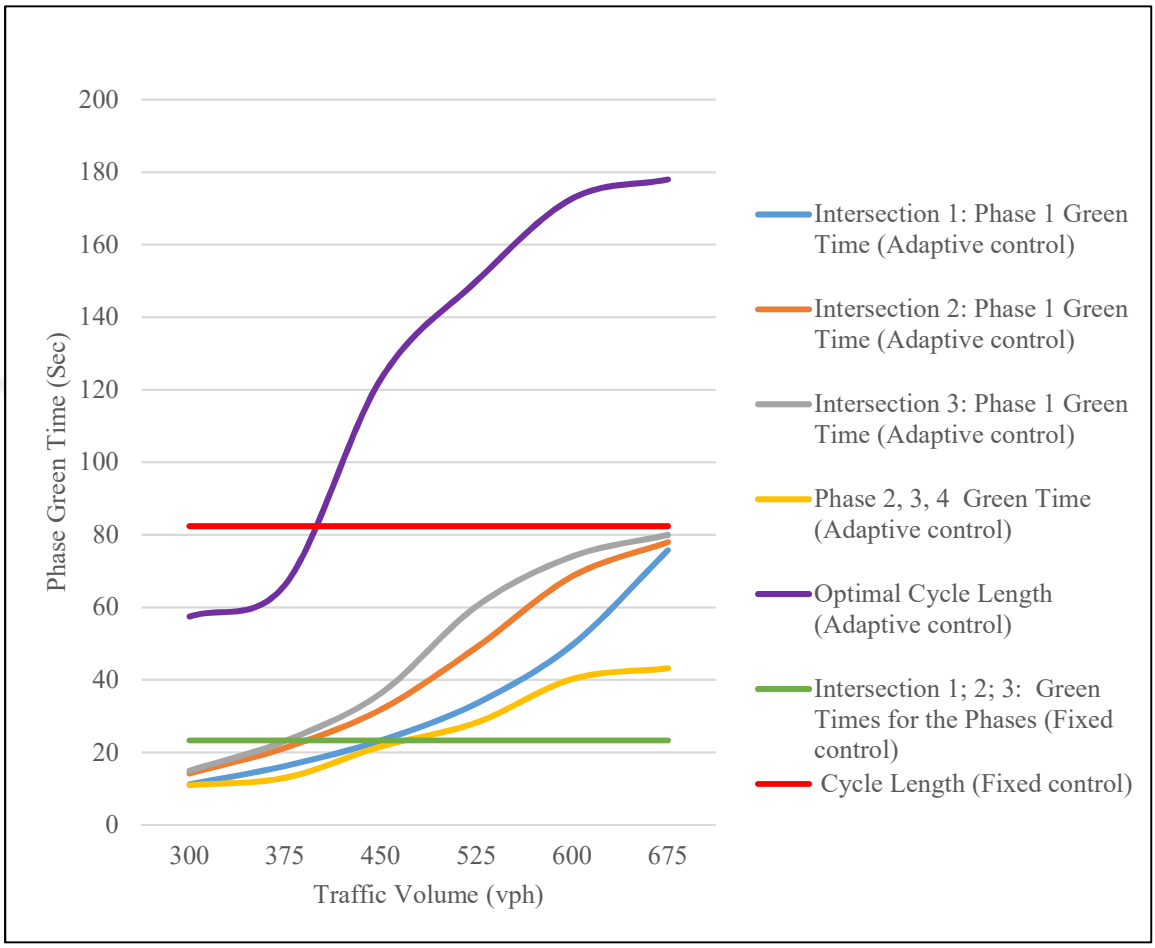

Figure 5. 8. Comparison of the Test results

5.2.2.3. Second Tests

In contrast to the first experimental tests, in these tests the controllers computed the phase green times with low traffic volume. For this, all formulas previously described were used. These were done in order to show why it is useless to apply a coordination to the traffic intersections or to the traffic networks with low traffic volume (low traffic demands). The optimal cycle length should be small, which will provide very low phase green time durations.

Table 5. 3. Test results for low traffic volume (Adaptive control)

Adaptive control					
Intersection 1 L1 Traffic Volume (vp5mins)	Phase 1 (L1,L2); (L1,L4) (L5,L6); (L5,L8)	Phase 2 (L7,L2); (L7,L8); (L3,L4); (L3,L6)	Phase 3 (L1,L2); (L1,L4); (L1,L6)	Phase 4 (L5,L2); (L5,L6); (L5,L8)	Cycle length (in seconds)
0	**Green Time (in seconds)**	**Green Time (in seconds)**	**Green Time (in seconds)**	**Green Time (in seconds)**	0
15	2.58	1.72	1.72	1.72	35.73
30	4.03	1.34	1.34	1.34	36.06
45	5.03	1.12	12.26	12.26	36.39
60	5.82	0.97	0.97	0.97	36.72
75	6.48	0.86	8.16	8.16	37.07
Intersection 2 Traffic Volume (vp5mins)	Phase 1 (L4,L9); (L4,L11) (L12,L5); (L12,L13)	Phase 2 (L14,L5); (L14,L9); (L10,L11); (L10,L13)	Phase 3 (L4,L9); (L4,L11); (L4,L13)	Phase 4 (L12,L5); (L12,L9); (L12,L13)	Cycle length (in seconds)
15	2.30	1.84	1.84	1.84	35.82
30	3.07	1.48	1.48	1.48	36.14
45	4.71	1.26	1.26	1.26	36.48
60	5.51	1.10	1.10	1.10	36.82
75	6.19	0.99	0.99	0.99	37.16
Intersection 3 L11 Traffic Volume (vp5mins)	Phase 1 (L11,L15); (L11,L17); (L18,L12); (L18,L19)	Phase 2 (L20,L12); (L20,L15); (L16,L17); (L16,L19)	Phase 3 (L11,L15); (L11,L17); (L11,L19)	Phase 4 (L18,L12); (L18,L15); (L18,L19)	Cycle length (in seconds)
15	2.11	1.97	1.97	1.97	35.90
30	4.41	2.06	2.06	2.06	36.23
45	6.89	2.14	2.14	2.14	36.57
60	9.55	2.23	2.23	2.23	36.91
75	12.39	2.31	2.31	2.31	37.26

Table 5. 4. Test results for low traffic volume (Fixed-time control)

Fixed Time Control					
Intersection 1,2,3 L1 Traffic Volume (vp5mins)	Phase 1 (L1,L2); (L1,L4) (L5,L6); (L5,L8)	Phase 2 (L7,L2); (L7,L8); (L3,L4); (L3,L6)	Phase 3 (L1,L2); (L1,L4); (L1,L6)	Phase 4 (L5,L2); (L5,L6); (L5,L8)	Cycle length (in seconds)
0	**Green Time (in seconds)**	**Green Time (in seconds)**	**Green Time (in seconds)**	**Green Time (in seconds)**	0
15	1.99	1.99	1.99	1.99	35.95
30	1.99	1.99	1.99	1.99	35.95
45	1.99	1.99	1.99	1.99	35.95
60	1.99	1.99	1.99	1.99	35.95
75	1.99	1.99	1.99	1.99	35.95
90	1.99	1.99	1.99	1.99	35.95

5.2.2.4. Comparison of the second test results

Figure 5.9. depicts the comparison of the adaptive control and fixed-time control obtained in Table 5.3. and Table 5.4. In this figure, the x-axis of the graph represents the traffic volume (vehicle number) values, and the y-axis represents the phase green time values. It is also seen that for adaptive control, the phase green times (blue, orange and gray curves) increase when the traffic volumes (vehicle numbers) increase, while other phase green times (yellow curve) decrease. Also, the increase of the computed cycle length durations (purple curve) is observed. For fixed-time control, the phase green times (red curve) as well as the cycle length durations (green curve) always remain constant while the traffic volume (vehicle numbers) increase. In addition, we also observed how the optimal cycle length durations were small, which provided very low phase green time durations.

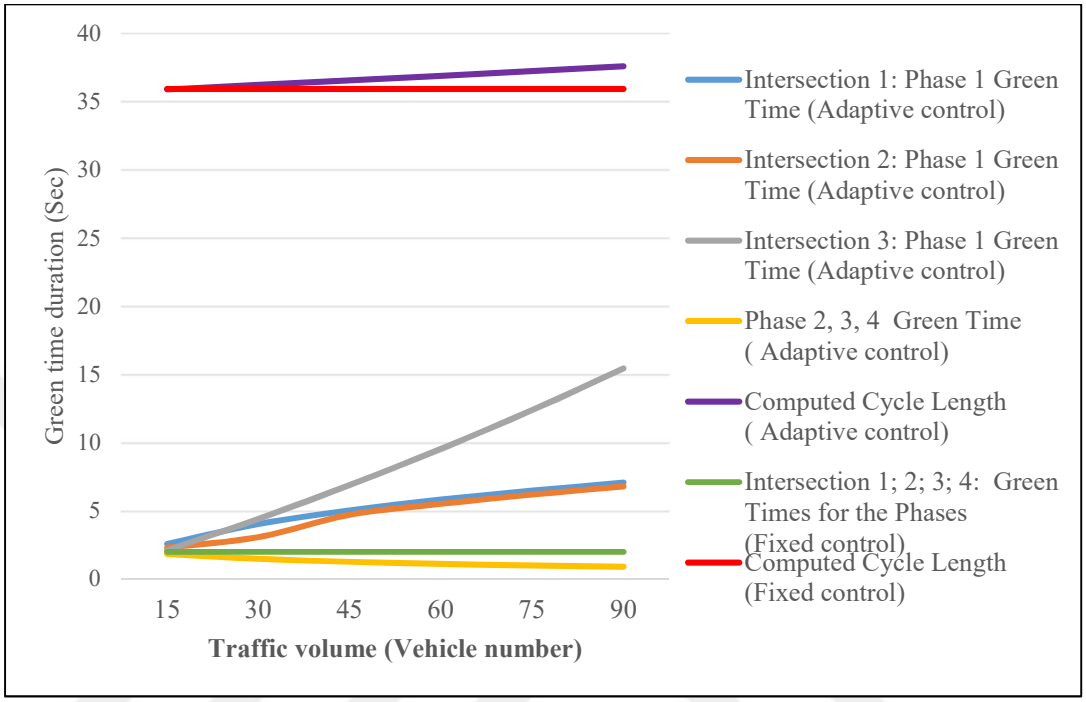

Figure 5.9. Comparison of the Test results

5.3. Result Comment and Summary

The observations made from the experimental test results can be summarized as follows. Firstly, as shown in Figure 5.22., the communication between the intersections was well established since the controllers displayed sent and received traffic data such as the cycle length, the road line traffic volume of the intersections, and phase green times without data loses during the transmission process.

Secondly, by using either high traffic volumes or low traffic volumes, the results show that for adaptive control, green time increase when the road line traffic volumes (vehicle numbers) increase. While for fixed-time control, green times and cycle remained constant when these traffic volumes increase.

Finally, it can be concluded that the proposed adaptive algorithm is better than fixed control algorithm because the controllers change timing plans every five minutes according to the dynamic variations of traffic volume (traffic demands or traffic flows). The obtained results were as expected and convinced that this system can be applied in real standard traffic network with three intersections. This system will efficiently contribute to the reduction of traffic congestion, therefore to the reduction of travel times and delays, fuel consumption, air pollution and road accidents at coordinated intersections.

6. CONCLUSIONS AND RECOMMENDATIONS

This chapter presents a conclusion concerning the entire study conducted in this thesis as well as the test results. It describes the recommendations based on the results obtained from experiment tests and on the developed algorithm. Moreover, it provides ideas for future research, which aims to improve the functionalities of the system proposed in this thesis.

6.1. Conclusions

An Adaptive Traffic Control (ATC) referred to an Intelligent Traffic Control (ITC) was designed to control an arterial traffic network with three intersections. This ATC included a coordinated traffic strategy, which was developed within a proposed algorithm. Moreover, In this algorithm was proposed a timing plan guideline including, (1) coordinated phase selection, (2) offset assignment, (3) phase sequence determinations, (4) split assignment, (5) optimum cycle length calculation, (6) effective green time calculation, (7) phase length calculation, and (8) transition mode definition. Also, a process of traffic volume comparison was proposed in order to assign the phase sequences according to the traffic volume (vehicle number) detected on the road lines. In order to analyze this algorithm, a three-intersection traffic prototype was implemented. Three Arduino board controllers were used to control these intersections, and a serial communication was established to provide the serial communication between the controllers. In addition, serial monitor commands from the Arduino software were programmed to display the traffic data transmitted by the intersections (controllers).

Experiments were carried out on the prototype in two stages. The first stage was to analyze the capacities of the controllers to transmit traffic data to each other and to display them. The second stage consisted of two cases, where the first case was to analyze the proposed algorithm. In this case, the tests were performed by increasing separately the traffic volume (vehicle numbers) of each road line every five minutes, in order to observe the variations of the cycle length and phase green time. The second was to analyze the prototype with a fixed-time algorithm. In this case, the tests were performed first by computing the cycle length and phase green time, which were maintained the same throughout the test period. Then, by increasing separately the traffic volume (vehicle number) of each road line every five minutes. Finally, by observing the effects of the traffic volume variations on both cycle length and phase green time.

The simulation results for the communication between the controllers showed a quick communication response. Also, it showed a complete display of sent and received traffic data such as the cycle length, the traffic volume of intersections, and phase green times without data loses during the transmission process.

The experimental test results from first case (adaptive traffic control) were benchmarked against the test results from second case (fixed-time traffic control). They showed the capacities of the proposed system in terms of its real-time responses according to the dynamic demands of traffic. These results were as expected and convinced that the system can be applied in real standard traffic network with three intersections. It will efficiently contribute to reduce traffic congestion, therefore to reduce travel times and delays, fuel consumption, air pollution and road accidents at coordinated intersections.

6.2. Recommendations

As observed in the test results, the phase green times increase as the detected road line traffic volume increases. This means that if a very high traffic volume is detected on a road line, a very high green time will be allocated to this line, which will necessarily lead to a

long waiting time on other road lines. The consequences would therefore be in terms of long queue of vehicles at intersections, and in terms of travel delay. So, it is recommended to limit the road line traffic volumes (number of vehicles) that can be detected at intersections to avoid these negative effects previously mentioned. The determination of these limits must be made after performing a serious study related to the traffic volume there is between the intersections that must be coordinated.

As shown in Figure 4.6. which presents the diagram of the algorithm, *TV* parameter was used for the transition process. So, it is recommended to use either a counter up or counter down in order to achieve this logical transition process. Example, if the *TV* parameter is set five minutes, a counter up must start the counting to zero, and the control system will update traffic data and compute new parameters when this counter up will reach the predefined five minutes. In case of counter down, the counting must start to five minutes, and the control system will update traffic data and compute new parameters when this counter down will reach zero.

The cycle chosen at the beginning of algorithm program must be in the range of 60 to 150 second in order to the system computes sufficient green lights times, which will allow the maximum vehicle number to pass through intersections. Example, depending on designer or engineer, the cycle length duration of 65, 75, 85, 100, or 120 seconds may be chosen at the starting of program.

6.3. Future Research

Future research will aim to improve the functionalities of the system proposed in this thesis. These improvements will focus on the development of a SCADA (Supervisory Control and Data Acquisition) application and on the implementation of a system based on connected vehicles, in which an algorithm for detecting and locating of emergency vehicles such as ambulances, firefighters and police will be developed.

We know that the dynamic control and remote monitoring of urban traffic flow from a center are becoming a necessity for municipal authorities in most cities around the world, due to the growth of traffic congestion at intersections. Thus, in contrast to the serial monitor commands programmed in this thesis to display traffic data transmitted between the intersections, the SCADA application will allow a real-time visualization of vehicle movements in traffic network. In addition, this application will remotely manage the traffic signal in case of malfunction of system components placed at traffic intersections. Connected vehicles will be equipped with a GPS module, and each vehicle will have a specific identity readable by smart sensors. A web page will be created in order to store information provided by the vehicle detected. The controllers will be able to receive information related to the identity and position of an emergency vehicle in order to calculate the duration of priority green signal that will be assigned to this vehicle to pass through the intersection without waiting or stopping.

REFERENCES

Al-Mudhaffar, A. (2006) *Impacts of Traffic Signal Control Strategies*, Ph.D. Thesis, Royal Institute of Technology, Stockholm, 224s.

Anonym, *Arduino board,* Tutorials Point Pvt. Ltd., New Delhi, *2016*, 223s.

Anonym, *Sustainable Urban Transportation System*, UNESCAP, CITYNET, Thailand, *2012, 49s.*

Arifin, M.F.B. (2014) *Development of Solar Powered Smart Traffic Light System*, B.Sc. Thesis, Universiti Teknikal Malaysia, Melaka, 64s.

Ashok, P.V., SivaSankari, S., Mani, V. and Sankaranarayanan, S. (2017) IoT Based Traffic Signaling System, *International Journal of Applied Engr Research,* 12(19): 8264-8269.

Aslania, M., Mesgaria, M.S. and Wieringb, M. (2018). Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events, *Elsevier Ltd*., 85: 732-752.

Badjie, O., Khalid, O. and Ali, O.M. (2016) *Traffic Signal Control Using Programmable Logic Controller (PLC)*, B.Sc. Thesis, Islamic University of Technology, Dhaka, 42s.

Baichtal, J. (2014*) Arduino for Beginners: Essential Skills Every Maker Needs*, 2. Pearson Education, USA, 57s.

Bonneson, J., Sunkari, S., Pratt, M. and Songchitruksa, P. (2009) *Traffic Signal Operations Handbook*, FHWA, Texas, 162s.

Davol, A.P. (2001) *Modeling of Traffic Signal Control and Transit Signal Priority Strategies in a Microscopic Simulation Laboratory*, M.Sc. Thesis, Brown University, Providence, 118s.

Eme, O., Madubuike, C.E., Idemudia, J.O. and Ntuen, A.U. (2016) Simulation of N-Way Traffic Lights Using Arduino Uno Environment, *ijcatr*, 5(8): 543-550.

Fan, L. (2014) Coordinated Control of Traffic Signals for Multiple Intersections, *SciRes*, 5: 2042-2049.

Fitzgerald, S. and Shiloh, M. (2012) *The Arduino Projects Book*, Arduino LLC., Torino, 175s.

Gao, J., Shen,Y., Liu, J., Ito, M. and Norio S. (2017) Adaptive Traffic Signal Control: Deep Reinforcement Learning Algorithm with Experience Replay and Target Network, *Sc.Ni*, 52: 767–781.

Ghosh, S., Shaurya Kumar, S. and Ghosh, G. (2017) Density Based Traffic Light System, *IJARECE*, 6(6): 590-593.

Gordon, R.L. and Tighe, W. (2005) *Traffic Control Systems HandBook*, Federal Highway Administration, USA, 367s.

Jiao, P. (2016) *Dynamic green split optimization in intersection signal design for urban street network*, M.Sc. Thesis, Purdue University, West Lafayette,115s.

Hamad, K. and Abuhamda, H. (2015) Estimating Base Saturation Flow Rate for Selected Signalized Intersections in Doha, *Journal of Traffic and Logistics Eng*, 3(2): 168-171.

Ketabdari, M. (2013) *Analysis of Adaptive Traffic Control Systems and design of a Decision Support System for better choice*, M.Sc. Thesis, Polytechnic University of Milan, Milan, 220s.

Kouvelas, A., Aboudolas, K., Papageorgiou, M., and Elias B.K. (2014) A Hybrid Strategy for Real-time Traffic Signal Control of Urban Road Networks, *IEEE Transactions on Intelligent Transportation Systems*, 12 (3): 884-894.

Lalitha, K. and Pounambal, M. (2018) Density Based Traffic Management Using IoT, *IJPAM*, 120: 781-792.

Leong, L.V., Wan., H.W.I. and Mohd-Sadullah, A.F. (2005) Determination of Ideal Saturation Flow at Signalized Intersections under Malaysian Road Conditions, *Journal of Transport Sc Soc Malaysia,* 1: 26-37.

Li, M. and Mallat, L. (2018) *Health impacts of air pollution*, 42. SCOR, China, 40s.

McShane, C. (1999) The Origins and Globalization of Traffic Control Signals, *Sage Publications*, 25(3): 379-404.

Mfenjou, M.L, Abba-Ari, A.A., Abdou, W., Kolyanga and Spies, F. (2018) Methodology and trends for an intelligent transport system indeveloping countries, *Elsevier Inc*., 19: 96-111.

Mirchandani, P. and Head, L. (2001) A real-time traffic signal control system: architecture, algorithms, and analysis, *Transportation Research part*, 9: 415-432.

Mishra, S., Bhattacharya, D. and Gupta, A. (2018) Congestion Adaptive Traffic Light Control and Notification Architecture Using Google Maps APIs, *data and mpdi*, 4(4): 83-90.

MnDOT. (2017) *Traffic Signal Timing and Coordination Manual*, Minnesota Department of Transportation road and travel information, USA, 297s.

Munem, A.S.A. and Croock, M.S. (2016) Smart Traffic Light Control System for Emergency Ambulance, *IJARCET*, 5(8): 2247-2255.

NCHRP. (2015) *Signal Timing Manual*, 2. National Academy of Sciences, USA, Washington, 322s.

Negi, P. (2006) *Artificial Immune System Based Urban Traffic Control*, M.Sc. Thesis, Texas A&M University, San Antonio, 81s.

Oladimeji, T. and Oshevire, P. (2014) *Design and Implementation of a Four ways*

or Junction prototype crossroad traffic light control system, JAET, 1: 2348-2931.

OTM. (2001) *Traffic Signal,* 12. Publications Ontario, Canada, Ontario, 173s.

Rajavali, G., Sravani, Y., Raju, P., Mrudhulatha, G. and Appalnaidu, CH. (2017) Arduino Based Smart Roads Controlling System for Future Cities, ijates, 5(4): 375-378.

Rajsman, , M. and Horvat., R. (2014) Public Urban Passenger Transport as Important Factor in the Development of Cities, *Jtle*, 2(3): 172-175.

Robert, L., Gordon, P.E. and Warren-Tighe, P.E. (2005) Traffic *Control Systems Handbook*, FHWA, USA, 367s.

Safi, M.E. (2016) Smart Traffic light controller based on Microcontroller, *ijccce,* 16(1): 38-45.

Sayyed, S., Date, P., Gautam, R. and Bhandari, G. (2014) Design of Dynamic Traffic Signal Control System, *IJERT*, 3(1): 933-938.

Schwartz, M. (2014) *Arduino Networking*, Packt Publishing Ltd., UK, 187s.

Seneviratne, P. (2017) *Building Arduino PLCs: The essential techniques you need to develop Arduino-based PLCs*, Pradeeka Seneviratne, Sri Lanka, 191s.

Shah, S. and Shah, U. (2016) *Arduino Blink Blueprints*, 1. Packt Publishing Ltd., UK, 136s.

Sharma, I. and Gupta, P.K. (2015) Study of Automatic Traffic Signal System for Chandigarah, *IJESRT*, 4(7): 26-33.

Shelby and Gebhart, S. (2001) *Design and evaluation of real-time adaptive traffic signal control algorithms*, Ph.D. Thesis, University of Arizona, Arizona, 488s.

Summala, H. (2000) Brake Reaction Times and Driver Behavior Analysis, *Transportation Human Factors,* 2(3): 217–226.

Smallwood, G., Schijns, S., Tedesco, M. and McCormick, M.O. (2001) *Traffic Manual*: *Traffic Signals*, 12. Publications Ontario, Ontario, 173s.

Smith, A., G. (2011) *Introduction to Arduino: A piece of cake*, 1. Alan G. Smith, USA, 172s.

Srinivasan, D., Choy, M.C. and Cheu, R.L. (2006) Neural Networks for Real-Time Traffic Signal Control, *IEEE,* 7(3): 261-271.

Thorne-Lyman, A., Jeff Wood, J., and Sam Zimbabwe, S. (2000) *Transit-Oriented Development Strategic Plan*, Metro TOD Program, 84s.

TRB. (2016) *Highway Capacity Manual*, 4. National Academy of Sciences, USA, Washington, 154s.

Valhavankar, S.N., Vibhute, A.S., Bhagat, A.G. and Said, S.K. (2016) Intelligent Traffic Control System: Emergency Vehicle Clearance & Lost

Vehicle Detection, *IJIRCCE* 4(4): 6416-6423.

Varun, K.S., Kumar, K.A., Chowdary, V.R. and Raju, C.S.K. (2018) A Perceptive Model of Traffic Flow: Using Arduino Board, *iiste*, 72.

APPENDIX

Appendix A. Coding for the comparison function

Master__For_Communication_Simulation §

```
void Line_comparison() {
if(V1>V3 && V3>V5 && V5>V7){
gC=G1;gD=G2;gE=G3;gF=G4;
MovementC(); MovementD(); MovementE();MovementF();}
if(V3>V1 && V1>V5 && V5>V7){
gA=G1;gD=G2;gE=G3;gF=G4;
MovementA(); MovementD(); MovementE();MovementF();}
if(V5>V1 && V1>V3 && V3>V7){
gA=G1;gE=G2;gD=G3;gF=G4;
MovementA(); MovementE(); MovementD();MovementF();}
if(V7>V1 && V1>V3 && V3>V5){
gA=G1;gE=G2;gD=G3;gF=G4;
MovementA(); MovementE(); MovementD();MovementF();}
if(V1==V3 && V3==V5 && V5==V7{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V1==V3 && V3>V5 && V5>V7{
gC=G1;gD=G2;gE=G3;gF=G4;
MovementC(); MovementD(); MovementE();MovementF();}
```

Master__For_Communication_Simulation §

```
if(V5>V3 && V3>V1 && V1==V7{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V5>V7 && V7>V1 && V1==V5{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
```

Appendix A. (Continued)

Master__For_Communication_Simulation §

```
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V1==V7 && V7>V3 && V3>V5{
gC=G1;gF=G2;gD=G3;gE=G4;
MovementC(); MovementF(); MovementD();MovementE();}
if(V1>V3 && V3>V5 && V5==V7{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V1>V5 && V5>V3 && V3==V7{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V1>V7 && V7>V3 && V3==V5{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V3==V5 && V5>V7 && V7>V1{
gA=G1;gB=G2;gE=G3;gD=G4;
MovementA(); MovementB(); MovementE();MovementD();}
```

Master__For_Communication_Simulation §

```
if(V3==V7 && V7>V5 && V5>V1{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
if(V5==V7 && V7>V3 && V3>V1{
gA=G1;gB=G2;gE=G3;gF=G4;
MovementA(); MovementB(); MovementE();MovementF();}
if(V3>V1 && V1>V5 && V5==V7{
gA=G1;gB=G2;gD=G3;gE=G4;
MovementA(); MovementB(); MovementD();MovementE();}
if(V3>V5 && V5>V1 && V1==V7{
gA=G1;gB=G2;gD=G3;gE=G4;
MovementA(); MovementB(); MovementD();MovementE();}
if(V3>V7 && V7>V1 && V1==V5{
gA=G1;gB=G2;gD=G3;gE=G4;
MovementA(); MovementB(); MovementD();MovementE();}
if(V5>V1 && V1>V3 && V3==V7{
gA=G1;gB=G2;gA=G3;gB=G4;
MovementA(); MovementB(); MovementA();MovementB();}
```

Appendix B. Coding for the phase movements

Master__For_Communication_Simulation §

```
void MovementF()
{
  digitalWrite(Green1,LOW);     digitalWrite(Green1,LOW);
  digitalWrite(Green3,LOW);     digitalWrite(Green2,LOW);
  digitalWrite(Green5,LOW);     digitalWrite(Green3,LOW);
  digitalWrite(Green7,HIGH);    digitalWrite(Green4,LOW);
  digitalWrite(Yellow1,LOW);    digitalWrite(Yellow1,LOW);
  digitalWrite(Yellow3,LOW);    digitalWrite(Yellow3,LOW);
  digitalWrite(Yellow5,LOW);    digitalWrite(Yellow5,LOW);
  digitalWrite(Yellow7,LOW);    digitalWrite(Yellow7,HIGH);
  digitalWrite(Red1,LOW);       digitalWrite(Red1,LOW);
  digitalWrite(Red3,LOW);       digitalWrite(Red3,LOW);
  digitalWrite(Red5,LOW);       digitalWrite(Red5,LOW);
  digitalWrite(Red7,LOW);       digitalWrite(Red7,LOW);
  delay(gF);                    delay(YellowTime);
```

Master__For_Communication_Simulation §

```
void MovementE()
{
  digitalWrite(Green1,LOW);     digitalWrite(Green1,LOW);
  digitalWrite(Green3,LOW);     digitalWrite(Green2,LOW);
  digitalWrite(Green5,HIGH);    digitalWrite(Green3,LOW);
  digitalWrite(Green7,LOW);     digitalWrite(Green4,LOW);
  digitalWrite(Yellow1,LOW);    digitalWrite(Yellow1,LOW);
  digitalWrite(Yellow3,LOW);    digitalWrite(Yellow3,LOW);
  digitalWrite(Yellow5,LOW);    digitalWrite(Yellow5,HIGH);
  digitalWrite(Yellow7,LOW);    digitalWrite(Yellow7,LOW);
  digitalWrite(Red1,LOW);       digitalWrite(Red1,LOW);
  digitalWrite(Red3,LOW);       digitalWrite(Red3,LOW);
  digitalWrite(Red5,LOW);       digitalWrite(Red5,LOW);
  digitalWrite(Red7,LOW);       digitalWrite(Red7,LOW);
  delay(gE);                    delay(YellowTime);
```

Appendix B. (Continued)

Master__For_Communication_Simulation §

```
void MovementD()
{
  digitalWrite(Green1,LOW);     digitalWrite(Green1,LOW);
  digitalWrite(Green3,HIGH);    digitalWrite(Green2,LOW);
  digitalWrite(Green5,LOW);     digitalWrite(Green3,LOW);
  digitalWrite(Green7,LOW);     digitalWrite(Green4,LOW);
  digitalWrite(Yellow1,LOW);    digitalWrite(Yellow1,LOW);
  digitalWrite(Yellow3,LOW);    digitalWrite(Yellow3,HIGH);
  digitalWrite(Yellow5,LOW);    digitalWrite(Yellow5,LOW);
  digitalWrite(Yellow7,LOW);    digitalWrite(Yellow7,LOW);
  digitalWrite(Red1,LOW);       digitalWrite(Red1,LOW);
  digitalWrite(Red3,LOW);       digitalWrite(Red3,LOW);
  digitalWrite(Red5,LOW);       digitalWrite(Red5,LOW);
  digitalWrite(Red7,LOW);       digitalWrite(Red7,LOW);
  delay(gD);                    delay(YellowTime);
```

Master__For_Communication_Simulation §

```
void MovementC()
{
  digitalWrite(Green1,HIGH);    digitalWrite(Green1,LOW);
  digitalWrite(Green3,LOW);     digitalWrite(Green2,LOW);
  digitalWrite(Green5,LOW);     digitalWrite(Green3,LOW);
  digitalWrite(Green7,LOW);     digitalWrite(Green4,LOW);
  digitalWrite(Yellow1,LOW);    digitalWrite(Yellow1,HIGH);
  digitalWrite(Yellow3,LOW);    digitalWrite(Yellow3,LOW);
  digitalWrite(Yellow5,LOW);    digitalWrite(Yellow5,LOW);
  digitalWrite(Yellow7,LOW);    digitalWrite(Yellow7,LOW);
  digitalWrite(Red1,LOW);       digitalWrite(Red1,LOW);
  digitalWrite(Red3,LOW);       digitalWrite(Red3,LOW);
  digitalWrite(Red5,LOW);       digitalWrite(Red5,LOW);
  digitalWrite(Red7,LOW);       digitalWrite(Red7,LOW);
  delay(gC);                    delay(YellowTime);
```

Appendix B. (Continued)

Master__For_Communication_Simulation §

```
void MovementB()
{
  digitalWrite(Green1,LOW);     digitalWrite(Green1,LOW);
  digitalWrite(Green3,HIGH);    digitalWrite(Green2,LOW);
  digitalWrite(Green5,LOW);     digitalWrite(Green3,LOW);
  digitalWrite(Green7,HIGH);    digitalWrite(Green4,LOW);
  digitalWrite(Yellow1,LOW);    digitalWrite(Yellow1,LOW);
  digitalWrite(Yellow3,LOW);    digitalWrite(Yellow3,HIGH);
  digitalWrite(Yellow5,LOW);    digitalWrite(Yellow5,LOW);
  digitalWrite(Yellow7,LOW);    digitalWrite(Yellow7,HIGH);
  digitalWrite(Red1,LOW);       digitalWrite(Red1,LOW);
  digitalWrite(Red3,LOW);       digitalWrite(Red3,LOW);
  digitalWrite(Red5,LOW);       digitalWrite(Red5,LOW);
  digitalWrite(Red7,LOW);       digitalWrite(Red7,LOW);
  delay(gB);                    delay(YellowTime);
```

Master__For_Communication_Simulation §

```
void MovementA()
{
  digitalWrite(Green1,HIGH);    digitalWrite(Green1,LOW);
  digitalWrite(Green3,LOW);     digitalWrite(Green2,LOW);
  digitalWrite(Green5,HIGH);    digitalWrite(Green3,HIGH);
  digitalWrite(Green7,LOW);     digitalWrite(Green4,LOW);
  digitalWrite(Yellow1,LOW);    digitalWrite(Yellow1,HIGH);
  digitalWrite(Yellow3,LOW);    digitalWrite(Yellow3,LOW);
  digitalWrite(Yellow5,LOW);    digitalWrite(Yellow5,HIGH);
  digitalWrite(Yellow7,LOW);    digitalWrite(Yellow7,LOW);
  digitalWrite(Red1,LOW);       digitalWrite(Red1,LOW);
  digitalWrite(Red3,LOW);       digitalWrite(Red3,LOW);
  digitalWrite(Red5,LOW);       digitalWrite(Red5,LOW);
  digitalWrite(Red7,LOW);       digitalWrite(Red7,LOW);
  delay(gA);                    delay(YellowTime);
```

CIRRICULUM VITAE

Personal Information

Name and Surname : Martinien Gloire Kanguet

Date of Birth : 14/07/1990

Nationality : Kongo

Marital Status : Single

Phone : +90 552 206 7922

E-mail : kanguetmartinien22@gmail.com

Education

Degree	School / University	Year
High School	Kongo Technical High School	2012
Bachelor's Degree	Burkina Faso Graduate School of Electrical Engineering	2015
Master of Science	Muğla Sıtkı Koçman University/ Electrical and Electronics Engineering	2017

Work Experience

Year	Company / City	Position
2017	National Electricity Company of Burkina Faso/Ouagadougou	Senior technician of electrical networks control
2015	National Electricity Company of Burkina Faso/Ouagadougou	Senior technician of electrical networks control
2014	National Electricity Company of Burkina Faso/Ouagadougou	Maintenance Technician of Electrical Distribution Networks

Foreigh Languages

Dil (İngilizce, vs)	Beginner	Intermediate	Upper Interme-diate
English			X
French			X
Turkish	X		

Hobbies

1. Football
2. Reading
3. Cooking