SN Computer Science (2021) 3:68
https://doi.org/10.1007/s42979-021-00956-3

ORIGINAL RESEARCH

Adaptive Fuzzy-Logic Traffic Control Approach Based on Volunteer IoT Agent Mechanism

Guan Hewei[1] · **Ali Safaa Sadiq**[2] · **Mohammed Adam Taheir**[3]

Received: 19 January 2021 / Accepted: 19 October 2021 / Published online: 18 November 2021

Abstract
Traffic congestion is an extremely common phenomenal issue; it occurs in many cities around the world, especially in those cities with high car ownership. Traffic congestion not only causes air pollution and fuel wastage, but it also leads to an increased commuting time and reduces the work time availability. Due to these reasons, traffic congestion needs to be controlled and reduced. The traffic light is the most widely adopted method to control traffic; however, most traffic lights in use are designed based on the pre-defined interval, which cannot cope with traffic volume change very well. Therefore, Internet of Things (IoT)-based traffic light or adaptive traffic light systems are developed in the recent years as a complement of the traditional traffic lights. The adaptive traffic light can be built based on monitoring current traffic situation or using vehicle-to-vehicle and vehicle-to-infrastructure communication. This paper proposes a new design of adaptive traffic light, this traffic light system is based on fuzzy logic, and it introduces for the first time the use of volunteer IoT agent mechanism, which introduces more accurate results. To note that the presented work in this paper is the extended version of the work presented in (Hewei et al. in Fuzzy-logic approach for traffic light control based on IoTtechnology, 2021).

Keywords Internet of Things · Traffic control · Fuzzy logic · Adaptive traffic light

Introduction

Traffic congestion refers to the phenomenon of a road traffic bottleneck that produced by heavy car's traffic that heading to congested intersections with a slow speed. This case is usually occurring during rush hours and holidays. This situation often occurs in major metropolitan areas around the world, areas with high automobile usage, and highways connecting two cities.

This article is part of the topical collection "Intelligent Computing and Networking" guest edited by Sangeeta Vhatkar, Seyedali Mirjalili, Jeril Kuriakose, P.D. Nemade, Arvind W. Kiwelekare, Ashok Sharma, and Godson Dsilva.

✉ Ali Safaa Sadiq
ali.sadiq@wlv.ac.uk

[1] School of Information Technology, Monash University, Subang Jaya, Malaysia

[2] School of Mathematics and Computer Science, University of Wolverhampton, Wulfruna Street, Wolverhampton WV1 1LY, UK

[3] Faculty of Technology Sciences, Zalingei University, Zalingei, Sudan

Frequent traffic congestion leads to an increased commuting time and it reduces the time available for work, resulting in economic losses due to workers' delay getting their working places on time, delay of delivering goods on time, and so many other factors. It also causes drivers to feel irritated and impatient, which increases their stress and further damages their health. To a certain extent, traffic congestion is also a waste of fuel and pollution: engines keep running during traffic congestions, which consume extra fuel, and at the time of congestion, drivers always accelerate and brake back and forth, which further increases fuel consumption. Traffic congestion therefore not only wastes energy, but also causes air pollution.

To manage vehicles on the road more efficiently and to keep safety of both pedestrians and vehicles, a contraption was put up on a busy thoroughfare in the UK in London, England in 1868, this contraption was composed of a revolving lamp fixture, and the lanterns were managed by operators, and their lights were mainly powered by natural gas. This is known as the first traffic light in human history [4]. With the time goes by, industrial and manufacturing level has grown tremendously; the number of vehicles has also grown rapidly. According to a report released by research house Bernstein, the global number of cars on the road will double by 2040, and it will reach the 2 billion marks by that time [10].

Nowadays, the most widely adapted traffic management method is using traffic light in traffic junctions; in most of the time, traffic light can manage traffic orderly, but in some cases, traffic officers also have to manage traffic junction in person when traffic light cannot handle heavy traffic flow well. This is due to the fact that most of the traffic lights in use today are based on a pre-defined time interval. This time interval is derived from observation on the local traffic condition and expert experiences, which they can handle normal traffic flow well; but in the real life, traffic condition is not predictable. In other words, traffic condition is a random parameter and likely to change often. Therefore, a simple pre-defined traffic light interval is not enough to handle traffic situation perfectly in the real life.

Scientists and researchers have done many research studies to pursue a better solution to the problem. One of the possible solutions is the adaptive traffic light. An adaptive traffic light system employs techniques like image processing and sensor detecting to analyze current traffic flow dynamically, and then make a decision on how to adjust traffic light interval accordingly to optimize traffic efficiency in a traffic junction. Another possible solution is using inter-vehicle and inter-infrastructure communications; however, this method is less feasible, because it is very expensive to implement, as it enquires many hardware components to support its functionality.

In this paper, a design of adaptive traffic light is proposed, this traffic light system is based on fuzzy logic, and it introduces for the first time the use of volunteer IoT agent mechanism to make sure that the system gives a more accurate result, as introduced earlierin [8]. The proposed adaptive traffic light system will use sensors to monitor current traffic situation at a traffic junction and collect data like traffic saturation and queue length from the sensors. The system also takes volunteer IoT-based agent's feedback on traffic situation as another input. Inputs will be fed into a developed fuzzy logic control algorithm. The output of the system is a variable named Green Time, which indicates the duration of next traffic light green phase. We envisage that using our proposed system will facilitate traffic management unit in smart cities with an application for smart and adaptive traffic management mechanism. Our proposed system helps them in making automated and prompt responses to such dynamic traffic situation and mitigate possible congestion.

Related Work

There are many studies, which have examined different methods on traffic management. The authors of Ref. [5] proposed a traffic congestion detection system in their paper; their system consists of I2I (infrastructure-to-infrastructure) communication method, V2I (vehicle-to-infrastructure) communication method, V2V (vehicle-to-vehicle) communication method, and big data cluster. The system uses above-mentioned three communication method to collect all the vehicular information such as vehicle's latitude, vehicle's longitude, and vehicle's speed. Then, data are encapsulated in DATA packets using the LORA-CBF algorithm and transmitted to the big data cluster. In big data cluster, received data are interpreted and further analyzed using algorithm based on Binary Traffic Output (BTO) algorithm proposed by Gupta et al. [7] to identify potential traffic events such as traffic congestion and traffic accident. Similar work has been done in Milanes et al. [11]; this proposed system employs V2I communication technology based on WAVE IEEE 802.11p standard and uses fuzzy logic to coordinate vehicles on the road. Fuzzy control system considers each vehicle's comfortable and safe distance and speed adjustment to prevent collisions in advance and improve traffic flow. This traffic management system is capable of analyzing received information and sending warning or recommendation commands to the vehicles.

As mentioned above, the adaptive traffic light is a promising method to manage traffic flow due to its feasibility and relatively low cost. Compare adaptive traffic light to I2I, V2I, and V2V communication technologies; adaptive traffic light system requires fewer resources. In the above-mentioned three communication technologies, to achieve communication between vehicles and infrastructures, additional communication module or detection module is required to be installed in the vehicles, which is not feasible in the real life; there is an enormous number of vehicles in the real life. However, an adaptive traffic light system does not require these extra modules. Therefore, many researchers have research in this area.

In the article [1], authors built an IoT-based traffic junction model, with several ultrasonic sensors placed on sides of road to count the number of cars and connect sensors to Arduino microcontroller, and then, data are transmitted to Raspberry Pi3 microcontroller through Wi-Fi module, where analysis been carried out. The traffic was categorized as heavy and normal traffic, as well as based on the traffic density; traffic signal time has been changed accordingly. Azura et al. have conducted a research on adaptive traffic light system using MATLAB simulation; in their research paper [2], they developed a fuzzy traffic controller based on vehicles queue length and waiting time at current green phase. They compared fuzzy traffic controller (FTC) with vehicle-actuated controller (VAC), and found that the fuzzy controller distinctly reduces average waiting time, average queue length, and delay time. In another research done by authors of Ref. [12], it divides the status of intersections into four categories. Then, based on the evaluation of the status, it adopts a fuzzy reasoning method to evaluate the traffic operation status of intersections and formulate corresponding traffic management method for traffic congestion. Table 1 compares the differences between above-mentioned studies including our proposed method.

As we can see from the above table, most of the studies use fuzzy logic as a control algorithm. Fuzzy logic is one of Artificial Intelligence (AI) techniques aims at imitating the way human intelligence think and solve problems; it can handle situations where the available information is imprecise and vague [3]. Therefore, it is suitable for a traffic light control system.

Proposed Method

Since traffic congestion has many negative influences on people's daily life, the objective of this paper is to build a traffic light control system based on fuzzy logic. The system should be able to determine the duration of traffic light's green time based on given inputs, which are traffic saturation, queue length, and traffic status from volunteer IoT agent. The system will monitor traffic condition during current traffic light cycle (one traffic light cycle contains three different phases, green-light phase, yellow light phase, and red-light phase), and then, the system will adjust the duration of green-light phase in the next traffic light cycle. Eventually, this system can help to improve traffic efficiency in a traffic junction and manage traffic more effectively by adjusting traffic light's green time. Figure 1 shows how system's architecture at a traffic junction.

In Fig. 1, four sensors are deployed on the side of the road. Vehicles count sensor is placed right next to the traffic light. During the green-light phase, every time a car passes through the intersection the movement can be captured by this sensor. The total vehicle count is the actual traffic volume in the intersection, after getting the total vehicle count;

Table 1 Comparison of existing work crossed with proposed method

Reference number	Implementation scenario	Input parameters	Control algorithm
[5]	I2I, V2I, V2V communication	Time between iterations Speed thresholds Queue length thresholds	Algorithm based on BTO algorithm
[11]	V2I communication	Errors in the distance (actual and reference distances between vehicles), Errors in the speed (the actual and recommended speeds)	Fuzzy logic
[1]	Ultrasonic sensor	Car count	Based on number of cars, adjust traffic signal time
[2]	MATLAB simulation	Waiting time, Vehicles queue length at current green phase	Fuzzy logic
[12]	Simulation based on the real-world intersection	Traffic saturation, Queue length, Delay time	Fuzzy logic
Our proposed method	MATLAB simulation	Traffic saturation, Queue length, Traffic status from volunteer IoT agent	Fuzzy logic

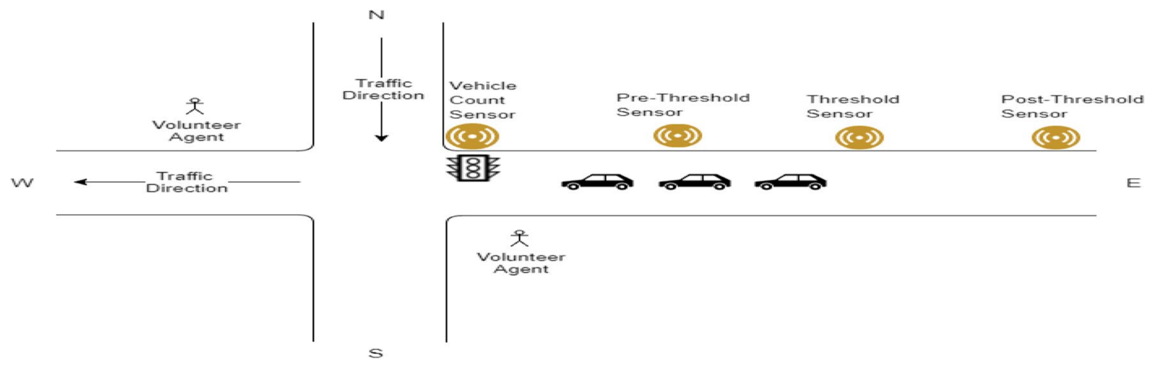


Fig. 1 Working scenario of the system

this value is then used to calculate traffic saturation, which is explained in the section "Traffic Saturation".

Pre-threshold, threshold, and post-threshold sensors are used to detect vehicle queue length during the red-light phase. If queue length ≤ 40 M, the pre-threshold sensor will be triggered. If (40 M < queue length < 60 M), the threshold sensor will be triggered. While, if the queue length ≥ 60 M, the post-threshold sensor will be triggered.

Also, in Fig. 1, there are two volunteer IoT-based agents at the intersection, volunteer agents can observe and evaluate traffic situation and send traffic situation value (e.g., minor congestion, server congestion) to the control system, and details of volunteer agent are discussed in the section "Traffic Status from Volunteer IoT Agent". In the real life, there can be multiple volunteer IoT agents in one intersection observe and send traffic situation data at the same time. However, this paper will assume only one volunteer IoT agent is allowed. Another assumption in the paper is that there are no turns involved, only straight through traffic, vehicles can only move from East to West or from North to South, as shown in Fig. 1.

Since the proposed method in this paper is built using simulation, all the inputs will be retrieved from a created input file; input file contains pre-defined possible values for inputs variables. Input files for each parameter are discussed in the section "Design of Fuzzy Inference-Based Traffic Saturation, Queue Length, and Traffic Status from Volunteer IoT Agent".

Design of Fuzzy Inference-Based Traffic Saturation, Queue Length, and Traffic Status from Volunteer IoT Agent

The fuzzy inference system (FIS) consists of fuzzification, knowledge base (membership functions and fuzzy rules), and defuzzification [6]. FIS takes crisp inputs and converts them into fuzzy inputs by using membership functions, and then, it combines fuzzy inputs together to come up with a fuzzy output based on fuzzy rules and output membership functions. Figure 2 [6] illustrates the workflow of fuzzy inference system in this paper.

Fuzzification of Inputs and Outputs

The fuzzy inference system takes three inputs, which are discussed in detail by the following sub-sections.

Traffic Saturation

This input metric indicating the degree of saturation of an intersection under traffic signal control is a measure of how much demand it is experiencing compared to its total capacity. Road saturation refers to the ratio of actual traffic volume and capacity of the road; it can be represented by x, and calculated using equation $x = \frac{q}{Q}$, where q is actual traffic volume and Q is total capacity. x can be any value between 0 and 1. If x is greater than 0.9, the traffic condition at the intersection deteriorated sharply, and traffic congestion is very likely to occur [12]. In this paper, the total capacity of the intersection is assumed to be 100 ($Q = 100$), which means during one green-light phase, the maximum number of vehicles to pass the intersection is 100. Membership function of traffic saturation is shown in Fig. 3. Y-axis indicates the degree of membership; X-axis indicates traffic saturation during the current green-light phase.

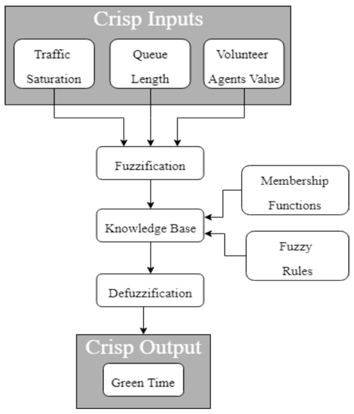


Fig. 2 Computation of output green time

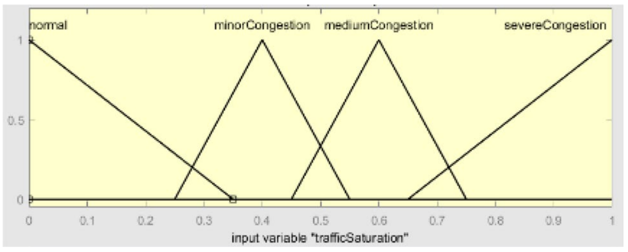


Fig. 3 Membership function for traffic saturation

Input file for traffic saturation is named Green Phase Car Count Input File; this file contains all the integer numbers from 0 to 100; in each green-light phase, system will randomly retrieve one number from the file as the actual traffic volume (e.g., $q = 50$) during that green-light phase, and then, q is substituted into equation $x = \frac{q}{Q}$ to calculate road saturation x.

Queue Length

The queue length input metric refers to the length of space occupied by the stopped vehicles; it can reflect the traffic flow at the intersection. It is of great significance for

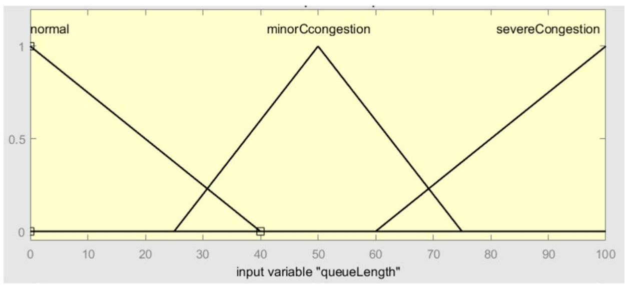


Fig. 4 Membership function for queue length

evaluating the operation status of the intersection, measuring the severity of traffic congestion, and evaluating the existing signal timing plan. In general, the more severe the congestion, the longer the length of the queue. The membership function of traffic saturation is shown in Fig. 4. *Y*-axis indicates the degree of membership; *X*-axis indicates queue length which is measured in meters.

As mentioned above, pre-threshold, threshold, and post-threshold sensors are used to detect vehicle queue length. If the pre-threshold value is triggered, it is considered as normal traffic condition; if the threshold is triggered, it is considered as minor congestion; if the post-threshold value is triggered, it is considered as severe congestion.

Input file for queue length is named Queue Length Input File, as indicated in Fig. 4, the minimum queue length is 0 m, and maximum queue length is 100 m. Therefore, this file contains all the integer numbers from 0 to 100; during each red-light phase, the system will randomly retrieve one number from the file as queue length during that red-light phase.

Traffic Status from Volunteer IoT Agent

Traffic information will also be gathered from volunteer agents, which could be any person from nearby traffic junction; they can observe and provide traffic situation information to the control system by mobile application. With the help of volunteer agents, the system takes account of nearby traffic junctions. For example, in case both current and next traffic junctions are having severe congestion, if the control system gives green light to current traffic junction, vehicles from current traffic junction will flood into next traffic junction and cause even worse traffic congestion. One solution can be extending the green time in the next junction to clear out vehicles, while current traffic junction can extend red time to hold vehicles, after the vehicles in the next junction have been cleared out, the current junction can open green light to let vehicles to pass. However, in this paper, we will assume that all volunteer agents come from current junction, which means that the system will only consider traffic situation at the current intersection.

Volunteer agent uses a scale of 1–5 as an indication of the traffic situation: 1 stands for no congestion, while 5 stands for severe congestion. Meanwhile, a trust level is assigned to each volunteer agent. Trust levels have values of high, medium, and low trust level; each trust level has its own weight, high = 20, medium = 15, and low = 10.

Calculation of a volunteer agent's value involves following steps:

Step 1: volunteer agent provides a number (e.g., 4) to indicate current traffic situation.

Step 2: system identifies volunteer agent's trust level (e.g., medium).

Step 3: multiply number provided in step 1 (in this case = 4) with the corresponding weight associated with trust level (in this case = 15), $4 \times 15 = 60$.

The result of step 3 (which is 60 in this case) is then used to evaluate traffic situation using membership function, as shown in Fig. 5.

Traffic status from volunteer agent has two input files; one file is called Volunteer Agent Value Input File, which contains integer numbers from 1 to 5; these numbers are used by volunteer agent as a scale to indicate traffic situation. Another file is called Volunteer Agent Trust Level Input File, since there are three trust levels, high, medium, and low. Thus, this file contains 3 numbers 20, 15, and 10; each number represents high trust level, medium trust level, and low trust level respectively. During each green-light phase, the system will randomly choose one number from each file to calculate volunteer agent's value.

The output of the system is a variable called GreenTime, which indicates how long the green traffic light will glow on next traffic light cycle. Membership function for output is shown in Fig. 6.

In the above figure, *X*-axis indicates the duration of the next green-light phase, which is measured in seconds, and maximum duration is 120 s.

Fuzzy Inference Engine

This section explains how fuzzy rules are derived. Fuzzy inference engine consists of a group of rules. There are three input parameters in this paper; both traffic saturation and

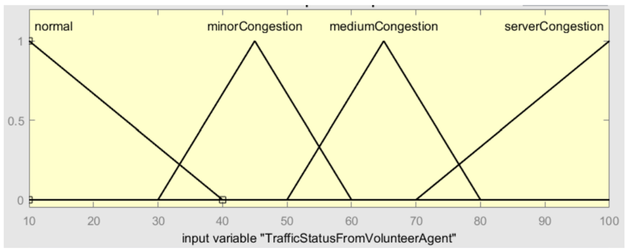


Fig. 5 Membership function for traffic status from volunteer agent

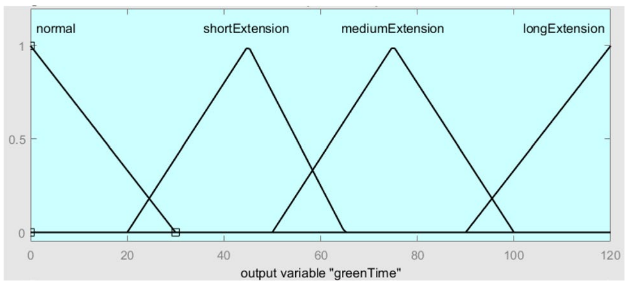


Fig. 6 Membership function for green time

Table 2 Classification of fuzzy rules

Pattern number	Input value	Output
1	3 normals	Normal
2	3 minors	Short extension
3	3 servers	Long extension
4	2 normals, 1 minor	Normal
5	2 normals, 1 medium	Short extension
6	2 normals, 1 server	Short extension
7	2 minors, 1 normal	Short extension
8	2 minors, 1 medium	Medium extension
9	2 minors, 1 server	Medium extension
10	2 mediums, 1 normal	Short extension
11	2 mediums, 1 minor	Medium extension
12	2 mediums, 1 server	Medium extension
13	2 servers, 1 normal	Long extension
14	2 servers, 1 minor	Long extension
15	2 servers, 1 medium	Long extension
16	1 normal, 1minor, 1 medium	Short extension
17	1 normal, 1minor, 1 server	Medium extension
18	1 normal, 1 medium, 1 server	Medium extension
19	1 minor, 1 medium, 1 server	Medium extension

traffic status from volunteer agent have four membership functions, so they both have four possible values: normal, minor congestion, medium congestion, and server congestion. The other input, queue length, has three membership functions, so it has three possible values: normal, minor congestion, and server congestion. Therefore, there are $4 \times 4 \times 3 = 48$ rules in total.

First, since all the inputs have the same weight, and order of inputs does not matter, thus, 48 rules can be classified into 19 different patterns, each pattern is assigned with an output value as shown in Table 2. This table is then used to identify output for each rule. Note that Table 2 lists only 19 combinations of the total of 48 rules just for the demonstration purposes.

According to Table 2, assign a proper output value to each rule, rules belong to the same pattern should have the same output. For example, rule No. 20 in Table 3, its first and second inputs have the value of minor congestion, its third input has the value of server congestion, and rule No. 22, as shown in Table 3, its first and third inputs have the value of minor congestion, and its second input has the value of server congestion. Both rules belong to pattern No. 9, because they both have two minor congestion inputs and one server congestion input. Thus, their outputs are both medium extensions. Forty-eight rules with clearly defined inputs and output and the first 22 rules are listed in Table 3 for the demonstration purposes. The complete set of rules were listed in Appendix A.

Based on the above-defined rules and membership functions, Figs. 7, 8 and 9 depict the correlation behavior between the input and output variables.

From the above figures, we can see that when both inputs variables indicate a normal traffic condition or minor congestion; the output variable Green Time stays in a very low level, which is the normal or short extension. However, as both input variables' value increase, the output variable's value will also increase gradually, and it will reach its peak when both input variables indicate server traffic congestion.

Defuzzification

Defuzzification is the process of transforming the fuzzy output obtained by the inference engine into a crisp value. In this paper, centroid defuzzification method is used. Centroid method returns the center of the area under the fuzzy set obtained from fuzzy inference engine.

Internal Design

Figure 10 shows internal design of the whole system.

Adaptive traffic light system consists of a main class, a fuzzy inference system, an interface file, a get input class, and four input files. Main class controls traffic light cycles and retrieves input data from input files using data retrieving method in getInput class, and feed data, which are traffic saturation, queue length, and traffic status from volunteer agent, into fuzzy inference system, fuzzy inference system will return an output which is green time, and then, main class will update user interface according to green time.

To further explain system's workflow, Fig. 11 is used to illustrate how system gets inputs and calculate output values, and how output values are used in the traffic light cycles.

As we can see from Fig. 11, in the 1st intersection traffic light cycle. During the North–South (NS) green time, the system will get NS traffic saturation data and NS traffic status from volunteer agent. Since NS is green, East–West (EW) should be red, and vehicles queue up during red time, so during NS green time, system also gets EW queue length data. Same things happen during the EW green time; system will get EW traffic saturation data, EW traffic status from volunteer agent, and NS queue length data.

Table 3 Knowledge structure based on fuzzy rules

Rule	Traffic saturation (input)	Queue length (input)	Traffic status from volunteer agent (input)	Green time (output)
1	Normal	Normal	Normal	Normal
2	Normal	Normal	Minor congestion	Normal
3	Normal	Normal	Medium congestion	Short extension
4	Normal	Normal	Severe congestion	Short extension
5	Normal	Minor congestion	Normal	Normal
6	Minor congestion	Normal	Normal	Normal
7	Minor congestion	Normal	Minor congestion	Short extension
8	Minor congestion	Normal	Medium congestion	Short extension
9	Minor congestion	Normal	Severe congestion	Medium extension
10	Minor congestion	Minor congestion	Normal	Short extension
11	Medium congestion	Normal	Normal	Short extension
12	Medium congestion	Normal	Minor congestion	Short extension
13	Medium congestion	Normal	Medium congestion	Short extension
14	Medium congestion	Normal	Severe congestion	Medium extension
15	Medium congestion	Minor congestion	Normal	Short extension
16	Medium congestion	Severe congestion	Severe congestion	Long extension
17	Severe congestion	Normal	Normal	Short extension
18	Severe congestion	Normal	Minor congestion	Medium extension
29	Severe congestion	Normal	Medium congestion	Medium extension
20	Severe congestion	Normal	Severe congestion	Long extension
22	Severe congestion	Minor congestion	Normal	Medium extension

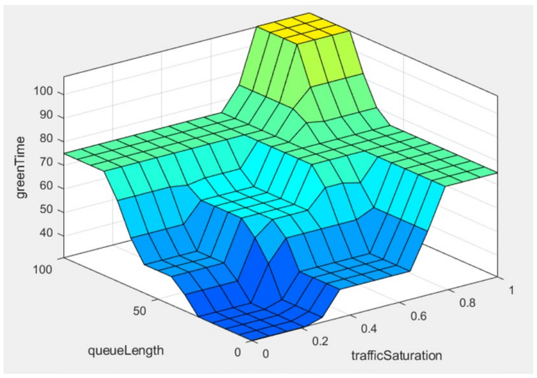


Fig. 7 Correlation between inputs (queue length and traffic saturation) and output fuzzy variables

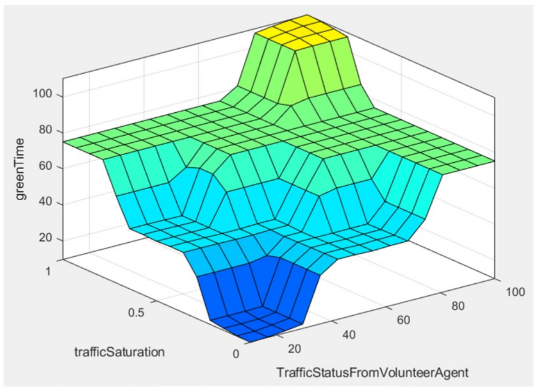


Fig. 8 Correlation between inputs (traffic saturation and traffic status from volunteer agent) and output fuzzy variables

The system can only start calculating output Green Time (within the range between 1 and 100 s as shown in Figs. 8, 9, 10), after getting all the above-mentioned input data, which means that the system will need two green times to get all the required input data. Therefore, first two green times are set to 10 s for initializing purpose. After retrieving data, fuzzy inference system returns output *m* which is the duration of NS green time, and *n* which is the duration of EW green time, and then, *m* and *n* are fed into the 2nd intersection traffic light cycle. In the 2nd intersection traffic light cycle, NS green light will glow for *m* seconds, and EW green light will glow for *n* seconds, data retrieving process is the same as the 1st intersection traffic light cycle, and fuzzy inference system returns new m and n values, then feed them into 3rd intersection traffic light cycle, and process of retrieving data and calculating output will be repeated.

External Design

A user interface has also been developed for visualizing purpose. With the help of simulation, people can better

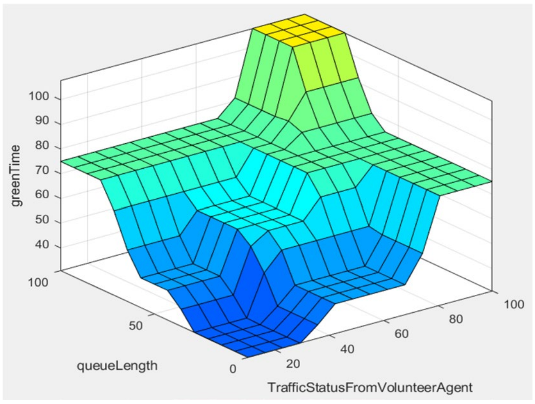


Fig. 9 Correlation between inputs (queue length and traffic status from volunteer agent) and output fuzzy variables

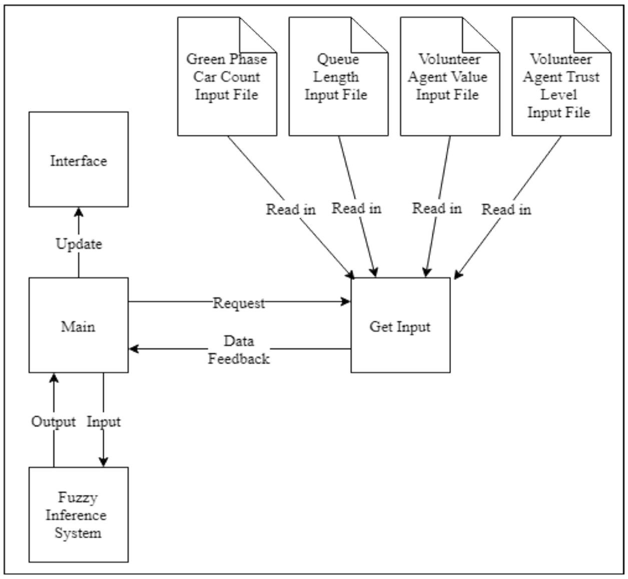


Fig. 10 System internal design

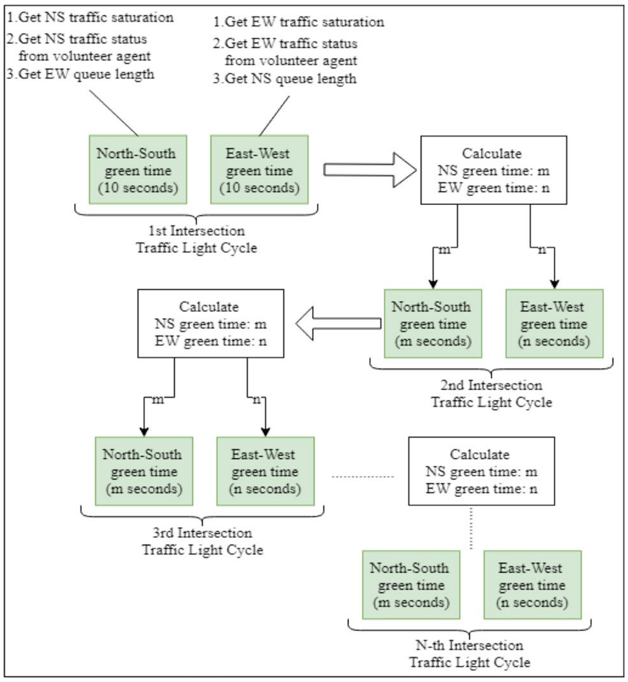
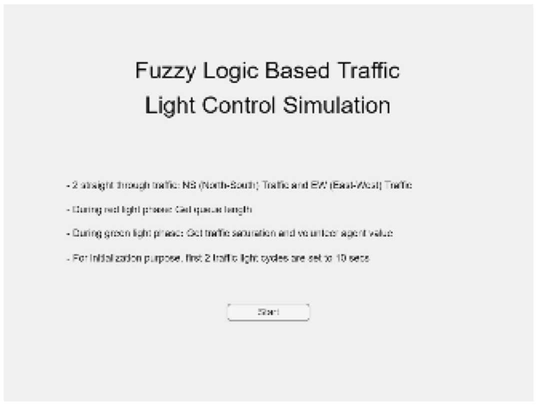

1.Get NS traffic saturation
2.Get NS traffic status from volunteer agent
3.Get EW queue length
1.Get EW traffic saturation
2.Get EW traffic status from volunteer agent
3.Get NS queue length
North-South green time (10 seconds)
East-West green time (10 seconds)
1st Intersection Traffic Light Cycle
Calculate NS green time: m EW green time: n
m
n
North-South green time (m seconds)
East-West green time (n seconds)
2nd Intersection Traffic Light Cycle
Calculate NS green time: m EW green time: n
m
n
North-South green time (m seconds)
East-West green time (n seconds)
3rd Intersection Traffic Light Cycle
Calculate NS green time: m EW green time: n
North-South green time (m seconds)
East-West green time (n seconds)
N-th Intersection Traffic Light Cycle

Fig. 11 Inputs retrieving and output deriving process



Fuzzy Logic Based Traffic
Light Control Simulation

- 2 straight through traffic: NS (North-South) Traffic and EW (East-West) Traffic
- During red light phase: Get queue length
- During green light phase: Get traffic saturation and volunteer agent value
- For initialization purpose, first 2 traffic light cycles are set to 10 secs

Fig. 12 Starting screen

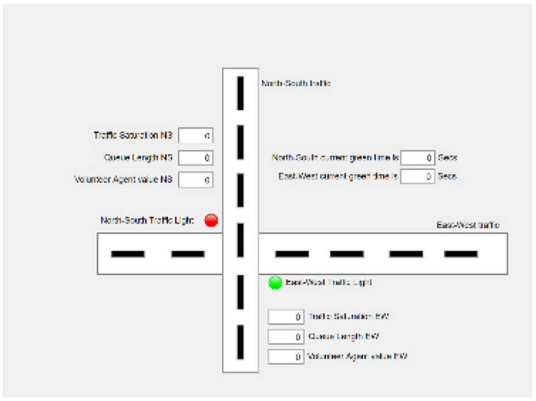
understand how the system works and see what is happening right now.

Figure 12 is the front screen of the system, which is representing the main entrance of the program; starting screen shows the method's title and some explanations about the proposed technique and its parameter settings. When the Start button is clicked, the callback function of the Start button will be invoked, new simulation screen will be showing, and the system will start running.

Figure 13 is the simulation screen before the program starts running; there are two traffic direction, North–South traffic and East–West traffic. Each traffic direction has its own traffic light, which are North–South Traffic Light and East–West Traffic Light. Traffic Saturation NS/EW, Queue Length NS/EW, and Volunteer Agent Value NS/EW text boxes are used to display values of input data. Duration of the current traffic light is displayed on the top right.

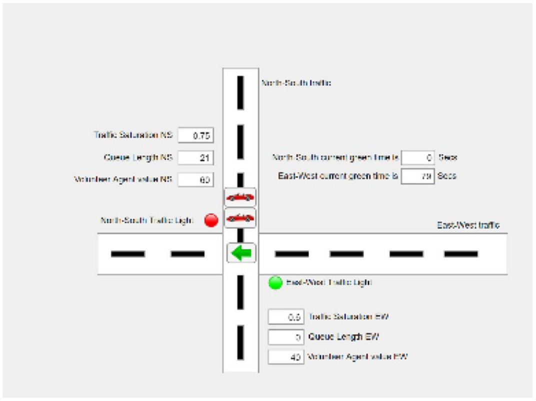
Figure 14 shows what will the simulation screen look like after the program is running and the pre-threshold sensor is triggered. A green arrow is showing to indicate the current green-light direction, which is East–West traffic, so the duration of East–West green light is displayed on the top right; this duration is calculated based on Traffic Saturation EW, Queue Length EW, and Volunteer Agent Value EW; however, since currently East–West direction is green light; thus, Queue Length EW is set to zero.

On the red-light side, images of vehicles are showing to indicate vehicle queue, and next duration of North–South



Traffic Saturation NS 0
Queue Length NS 0
Volunteer Agent value NS 0
North-South current green time is 0 Secs
East-West current green time is 0 Secs
0 Traffic Saturation EW
0 Queue Length EW
0 Volunteer Agent value EW

Fig. 13 Simulation screen before program running



Traffic Saturation NS 0.75
Queue Length NS 21
Volunteer Agent value NS 60
North-South current green time is 0 Secs
East-West current green time is 79 Secs
0.5 Traffic Saturation EW
5 Queue Length EW
40 Volunteer Agent value EW

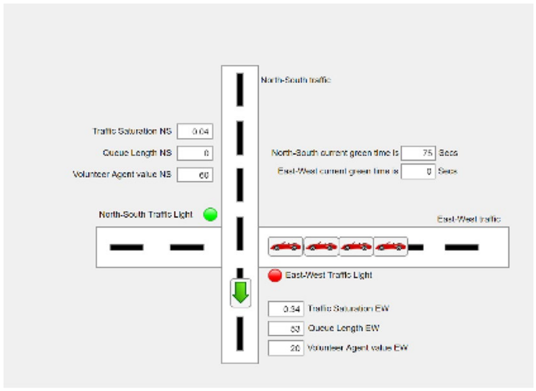
Fig. 14 Simulation screen with pre-threshold sensor triggered

green light will be calculated based on current Traffic Saturation NS (0.75), Queue Length *NS* (21), and Volunteer Agent Value NS (60).

If the pre-threshold sensor is triggered, two vehicles will be shown, if the threshold sensor is triggered, four vehicles will be shown; if the post-threshold sensor is triggered, six vehicles will be shown. Figures 15 and 16 show the case when the threshold and post-threshold sensors are triggered.

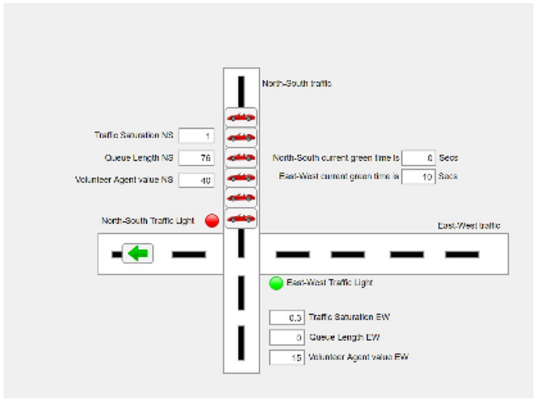
Results

There are many ways to evaluate performance and effectiveness of traffic light control system, such as observing how queue length has reduced over traffic light cycles, calculating the waiting time of vehicles, and measuring vehicle speed



Traffic Saturation NS 0.04
Queue Length NS 0
Volunteer Agent value NS 60
North-South current green time is 75 Secs
East-West current green time is 0 Secs
0.34 Traffic Saturation EW
53 Queue Length EW
20 Volunteer Agent value EW

Fig. 15 Simulation screen with threshold sensor triggered



Traffic Saturation NS 1
Queue Length NS 76
Volunteer Agent value NS 40
North-South current green time is 0 Secs
East-West current green time is 10 Secs
0.3 Traffic Saturation EW
0 Queue Length EW
15 Volunteer Agent value EW

Fig. 16 Simulation screen with post-threshold sensor triggered

as they drive through the intersection. Method of observing queue length and see if it is reduced is not feasible in this paper, because main control algorithm of this paper retrieves new queue length data in every traffic light cycle, which means that two consecutive queue length data are independent on each other, and current queue length does not have any relationship with its prior queue length data, so we cannot tell whether the queue length has reduced or not. Measuring vehicle speed is not feasible in this paper neither, because this developed simulation does not support measuring vehicle speed.

Therefore, this paper evaluates performance and effectiveness of traffic light control system by calculating average waiting time of vehicles, waiting time is how long vehicles have to wait before they can pass through the intersection. To calculate average waiting time, two parameters are required, one is the number of cars and another one is total waiting time.

To get the number of cars, assume the length of a car is 5 m, and then, we used the current queue length divide by 5 to obtain the possible number of cars in traffic light's queue.

Total waiting time can be calculated by adding together all the green time and time used to switch from green to red, which is the duration of yellow light, each time a traffic light switches from green to red, yellow light will glow for 2 s. Therefore, East–West total wait time = North–South total green time + 2, and North–South total wait time = East–West total green time + 2.

Executing the program for 500, 1000, 1500, 2000, 2500, 3000, 3500, and 4000 traffic light cycles to see system's performance in both short-term and long-term condition. The results of the testing were captured for 30 runs, and the average was taken for each individual scenario and is shown in Fig. 17. The statistically analysis proves the robustness of the proposed approach in maintaining the average waiting time using our proposed adaptive traffic light system.

As we can see from Fig. 17, both East–West and North–South average waiting time vary between 6.1 and 6.7 s. The longest waiting time in East–West traffic is 6.64 s, while in North–South traffic, longest waiting time is 6.58 s, and the shortest waiting time in East–West and North–South traffic are 6.20 and 6.18 s respectively. Although the number of traffic light cycles has increased a lot, the average waiting time does not change too much, so we can say that the number of traffic light cycles does not affect average waiting time.

To quantitatively evaluate the testing result, the Highway Capacity Manual 2010 [13] is used as a comparison, and HCM 2010 is a publication of the Transportation Research Board of the National Academies of Science in the United State, which has categorized signalized intersections to different Level of Service (LOS) by control Delay. LOS and control delay defined in this manual are summarized in Table 4 [13]. Control delay is the result of a control signal causes vehicles to reduce speed or to stop; the inherent concept of control delay is the same as waiting time. Therefore, control delay is used as a comparison with average waiting time.

Comparing the average waiting time per vehicle in the paper with HCM 2010 LOS criteria, we can see that all the testing results in Fig. 17 belong to LOS A group, which has control delay less than 10 s per vehicle. We can also see that testing result is stable; results only float with a small variation. Overall, the system shows a good performance and high efficiency.

More details on the running the developed adaptive fuzzy-based volunteer IoT agent mechanism for smart traffic light management system are presented in Appendix A.

Discussion and Recommendation

From the presented testing and obtained results in the previous section, we can see that, overall, the proposed adaptive traffic light control system in this paper shows a good performance, since all the testing results suggest Level of Service A. This is because fuzzy system makes good use of green traffic light by adjusting the duration of green light according to the current traffic situation, so the green-light time is never wasted, such case like green light is on, but there are actually no vehicles on the road will not happen. In other words, the system maximizes effective green time, and minimizes waiting time. Therefore, the system can show

Table 4 LOS criteria for signalized intersections

Level of service (LOS)	Control delay per vehicle (s/veh)
A	≤ 10
B	> 10–20
C	> 20–35
D	> 35–55
E	> 55–80
F	> 80

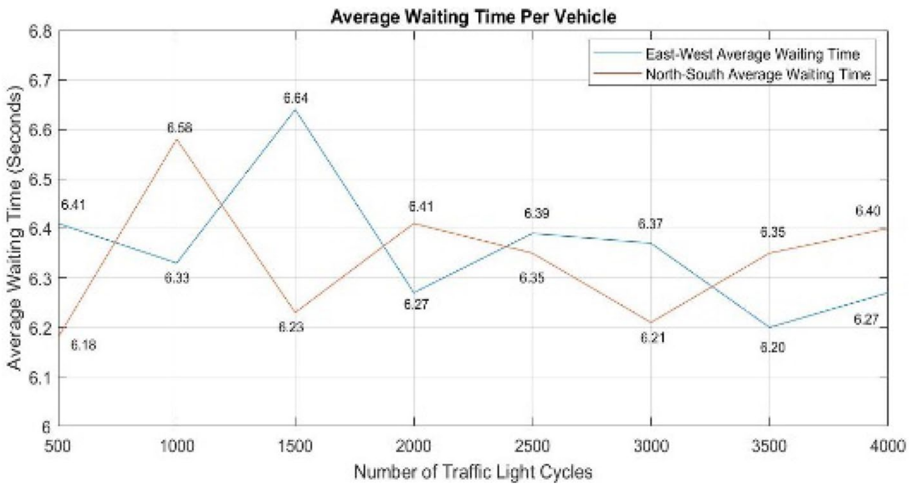


Fig. 17 Average waiting time per vehicle

a good performance in managing the traffic, which justifies the massive impact on the level of satisfaction by the traffic riders.

Although the system performs well, there are still several improvements can be done. The defined fuzzy inference system in this paper has proofed well enough the usefulness of the proposed method and shown the proof of concept has great potential, but it is still not perfect. Some changes can be made to the fuzzy membership functions, such as add more membership function, and define membership range base on more experiments and expert experiences, so the output surface can be smoother, and the result can be more accurate. Another improvement can be linking all the data together, which means that instead of introducing new input values in every traffic light cycle, the system can calculate new input values base on previous input values, so that the system and the output are more authentic. For example, the system can obtain the next queue length value by taking account of the current traffic volume, queue length, and vehicle speed.

In the future, this adaptive traffic light system can be used in a broader area. It can take account of traffic condition in the nearby traffic junctions, so several adaptive traffic light systems can collaborate together and manage traffic more effectively. The proposed method could be integrated to the smart cities' infrastructure to elevate traffic congestion points and avoid any traffic lights-related accidents. There are many installed CCTV's and other safely and security monitoring infrastructure in main cities, such as London and San Francisco (insider [9]), which can be used by our proposed system to be integrated for smart traffic light management. Besides, the proposed use of volunteer agents that can be easily integrated using smart mobile Apps, such as Google, associated application to report surrounded traffic status to be integrated to the proposed fuzzy inference system.

There are also some limitations that could be further improved in the future related to the sensitivity and scalability analysis of the proposed system considering interconnected huge traffic light system across relatively big cities.

Conclusion

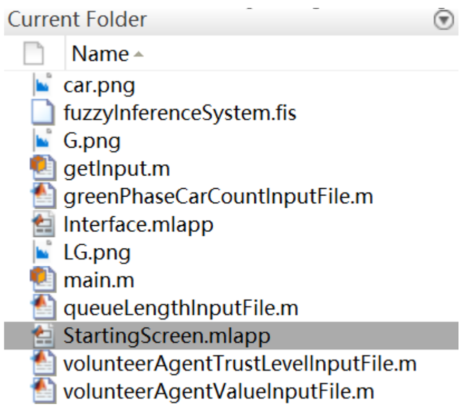
In this paper, a fuzzy logic-based adaptive traffic light system is proposed, and the signalized traffic model and fuzzy traffic controller have been developed using the MATLAB software. The best of our knowledge, this was the first-time volunteer agent concept which was integrated into a fuzzy-based inference system for managing smart traffic light system based on IoT networks. The effectiveness of the proposed method has been tested by running the system for 30 times and the statistical analysis has been taken for the obtained results to present the average waiting time using our proposed adaptive traffic light system. The results of the testing have shown that the system has a good performance using the proposed system that considered traffic saturation, queue length, and volunteer agent traffic behavior score value. Therefore, it is feasible and effective to use such adaptive traffic light system at a traffic junction in the smart cities.

As any other proposed method, there are some limitations as mentioned earlier, that is, the scalability and sensitivity of the proposed method should be further investigated and analyzed. For the future work, we are also planning to implement our proposed method to multi-interconnected traffic light networks to test the performance and study the possible optimization problem to be solved accordingly.

Appendix A

A. System deployment process

1. Open the project with MATLAB; after opening, following files will be showing:



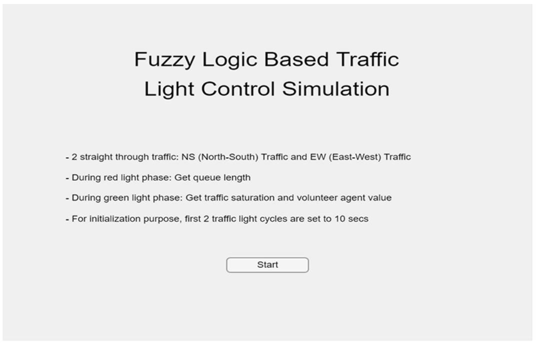
Current Folder

Name

car.png
fuzzyInferenceSystem.fis
G.png
getInput.m
greenPhaseCarCountInputFile.m
Interface.mlapp
LG.png
main.m
queueLengthInputFile.m
StartingScreen.mlapp
volunteerAgentTrustLevelInputFile.m
volunteerAgentValueInputFile.m

- fuzzyInferenceSystem.fis is fuzzy logic file.
- main.m is the MATLAB file contains main control code.
- getInput.m is MATLAB file contains function that reads data from input files.
- greenPhaseCarCountInputFile.m, queueLengthInputFile.m, volunteerAgentTrustLevelInputFile.m, volunteerAgentValueInputFile.m contain input values.
- Interface.mlapp is an MATLAB App Designer file contains a signalized traffic model.
- StartingScreen.mlapp is an MATLAB App Designer file, it is the main entrance of the program.

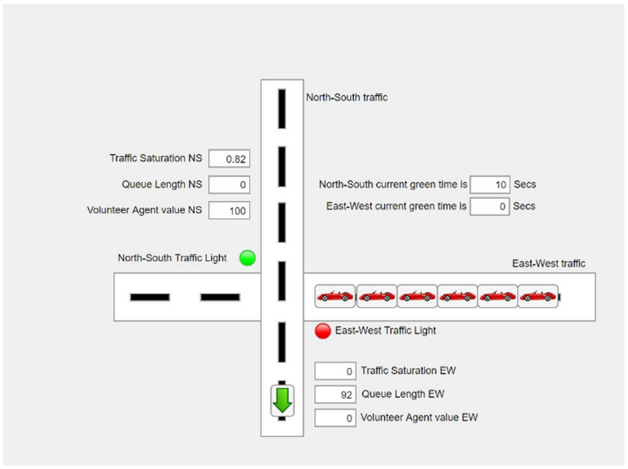
2. Run SetartingScreen.mlapp, and this screen will be showing:



Fuzzy Logic Based Traffic
Light Control Simulation

- 2 straight through traffic: NS (North-South) Traffic and EW (East-West) Traffic
- During red light phase: Get queue length
- During green light phase: Get traffic saturation and volunteer agent value
- For initialization purpose, first 2 traffic light cycles are set to 10 secs

3. Click the Start button on the previous screen, this will invoke callback function of Start button, and this new screen will be showing; the system will pause 2 s before start running:



Traffic Saturation NS 0.82
Queue Length NS 0
Volunteer Agent value NS 100
North-South current green time is 10 Secs
East-West current green time is 0 Secs
0 Traffic Saturation EW
92 Queue Length EW
0 Volunteer Agent value EW

4. Initially, the system is set to five traffic light cycles, means each traffic light will become green five times, if you want to change it, open main.m file, find the code in line 180, and change 5 to any other numbers.

B. Test report

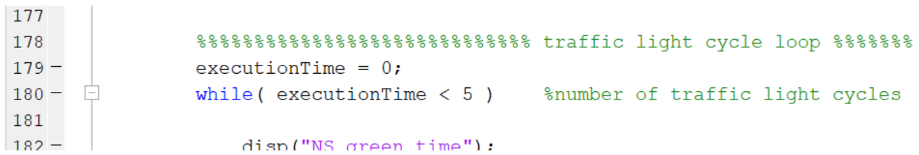
To test the correctness of fuzzy rules and fuzzy membership functions, open MATLAB fuzzy logic designer, from the menu bar select surface. By checking the surface of membership functions, as shown in Figs. 7, 8, and 9, the correctness of fuzzy rules and fuzzy membership functions can be tested, surface should be as smooth as possible. Then use evalfis function in MATLAB to evaluate fuzzy inference system.

To test performance and effectiveness of the program, execute system for 500, 1000, 1500, 2000, 2500, 3000, 3500, and 4000 traffic light cycles to see system's performance in both short-term and long-term condition. Before executing the system, some code needs to be added to the main.m file.

a) Two lines of code are added to calculate total waiting time:
NSTotalWaitTime = NSTotalWaitTime + EWGreenTime + 2;
EWTotalWaitTime = EWTotalWaitTime + NSGreenTime + 2;
b) Two lines of code are added to calculate total number of waiting vehicles:
EWTotalWaitCar = EWTotalWaitCar + (EWQueueLength/5);
NSTotalWaitCar = NSTotalWaitCar + (NSQueueLength/5);

After adding the code, execute system to get results. The results of testing are printed in Command Window; testing result is shown in the following table.

Total number of traffic light cycles	North–South total number of waiting cars	North–South total waiting time (s)	East–West total number of waiting cars	East–West total waiting time (s)
500	5140	31,784	4947	31,735
1000	9687	63,753	9953	63,037
1500	15,348	95,664	14,605	97,016



```
177
178          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%% traffic light cycle loop %%%%%%%
179 -        executionTime = 0;
180 -        while( executionTime < 5 )    %number of traffic light cycles
181
182 -            disp("NS green time");
```

Total number of traffic light cycles	North–South total number of waiting cars	North–South total waiting time (s)	East–West total number of waiting cars	East–West total waiting time (s)
2000	20,173	129,319	20,586	129,156
2500	25,054	159,222	24,854	158,944
3000	30,493	189,613	30,161	192,223
3500	35,052	222,457	35,520	220,333
4000	39,877	255,498	40,369	253,196

Table below lists the calculated average vehicle waiting time by dividing total waiting time by total number of waiting cars; final results are shown in this table:

Total number of traffic light cycles	North–South average waiting time (s)	East–West average waiting time (s)
500	6.18	6.41
1000	6.58	6.33
1500	6.23	6.64
2000	6.41	6.27
2500	6.35	6.39
3000	6.21	6.37
3500	6.35	6.20
4000	6.40	6.27

Table below lists the complete set of rules of the proposed fuzzy logic inference system.

Rule	Traffic saturation (input)	Queue length (input)	Traffic status from volunteer agent (input)	Green time (output)
1	Normal	Normal	Normal	Normal
2	Normal	Normal	Minor congestion	Normal
3	Normal	Normal	Medium congestion	Short extension
4	Normal	Normal	Severe congestion	Short extension
5	Normal	Minor congestion	Normal	Normal
6	Normal	Minor congestion	Minor congestion	Short extension
7	Normal	Minor congestion	Medium congestion	Short extension
8	Normal	Minor congestion	Severe congestion	Medium extension
9	Normal	Severe congestion	Normal	Short extension
10	Normal	Severe congestion	Minor congestion	Medium extension
11	Normal	Severe congestion	Medium congestion	Medium extension
12	Normal	Severe congestion	Severe congestion	Long extension
13	Minor congestion	Normal	Normal	Normal
14	Minor congestion	Normal	Minor congestion	Short extension
15	Minor congestion	Normal	Medium congestion	Short extension
16	Minor congestion	Normal	Severe congestion	Medium extension
17	Minor congestion	Minor congestion	Normal	Short extension
18	Minor congestion	Minor congestion	Minor congestion	Short extension
19	Minor congestion	Minor congestion	Medium congestion	Medium extension
20	Minor congestion	Minor congestion	Severe congestion	Medium extension
21	Minor congestion	Severe congestion	Normal	Medium extension
22	Minor congestion	Severe congestion	Minor congestion	Medium extension
23	Minor congestion	Severe congestion	Medium congestion	Medium extension
24	Minor congestion	Severe congestion	Severe congestion	Long extension
25	Medium congestion	Normal	Normal	Short extension
26	Medium congestion	Normal	Minor congestion	Short extension

Rule	Traffic saturation (input)	Queue length (input)	Traffic status from volunteer agent (input)	Green time (output)
27	Medium congestion	Normal	Medium congestion	Short extension
28	Medium congestion	Normal	Severe congestion	Medium extension
29	Medium congestion	Minor congestion	Normal	Short extension
30	Medium congestion	Minor congestion	Minor congestion	Medium extension
31	Medium congestion	Minor congestion	Medium congestion	Medium extension
32	Medium congestion	Minor congestion	Severe congestion	Medium extension
33	Medium congestion	Severe congestion	Normal	Medium extension
34	Medium congestion	Severe congestion	Minor congestion	Medium extension
35	Medium congestion	Severe congestion	Medium congestion	Medium extension
36	Medium congestion	Severe congestion	Severe congestion	Long extension
37	Severe congestion	Normal	Normal	Short extension
38	Severe congestion	Normal	Minor congestion	Medium extension
39	Severe congestion	Normal	Medium congestion	Medium extension
40	Severe congestion	Normal	Severe congestion	Long extension
41	Severe congestion	Minor congestion	Normal	Medium extension
42	Severe congestion	Minor congestion	Minor congestion	Medium extension
43	Severe congestion	Minor congestion	Medium congestion	Medium extension
44	Severe congestion	Minor congestion	Severe congestion	Long extension
45	Severe congestion	Severe congestion	Normal	Long extension
46	Severe congestion	Severe congestion	Minor congestion	Long extension
47	Severe congestion	Severe congestion	Medium congestion	Long extension
48	Severe congestion	Severe congestion	Severe congestion	Long extension

Declarations

Conflict of Interest We wish to confirm that there are no known conflict of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome. We confirm that the manuscript has been read and approved by all named authors, and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us. We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing, we confirm that we have followed the regulations of our institutions concerning intellectual property. We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions, and final approval of proofs.

References

1. Ashok P, SivaSankari S, Vignesh M, Suresh S. IoT based traffic signalling system. Int J Appl Eng Res. 2017;12(19):8264–9.
2. Azura C, Lai G, Haslina MdS. MATLAB simulation of fuzzy traffic controller for multilane isolated intersection. Int J Comput Sci Eng. 2010;02(04):924–33.
3. Baskar L, Hellendoorn J, De Schutter B, Papp Z. Traffic control and intelligent vehicle highway systems: a survey. IET Intell Transp Syst. 2011;5(1):38–52. https://doi.org/10.1049/iet-its.2009.0001.
4. Bernard M. Traffic light history abstracts. Academia; 2014.
5. Cárdenas-Benítez N, Aquino-Santos R, Magaña-Espinoza P, Aguilar-Velazco J, Edwards-Block A, Medina Cass A. Traffic congestion detection system through connected vehicles and big data. Sensors. 2016;16(5):599. https://doi.org/10.3390/s16050599.
6. Ghafoor K, Sadiq A, Abu Bakar K. A fuzzy logic approach for reducing handover latency in wireless networks. Netw Protoc Algorithms. 2011. https://doi.org/10.5296/npa.v2i4.527.

7. Gupta A, Choudhary S, Paul S. DTC: a framework to detect traffic congestion by mining versatile GPS data. In: 2013 1st international conference on emerging trends and applications in computer science. 2013. https://doi.org/10.1109/icetacs.2013.6691403.
8. Hewei G, Sadiq AS, Tahir MA. Fuzzy-logic approach for traffic light control based on IoT technology. In: Balas VE, Semwal VB, Khandare A (eds) Intelligent computing and networking: proceedings of IC-ICN 2021. Springer; 2021.
9. Insider Intelligence. How IoT and smart city technology works: devices, applications and examples. 2021. https://www.businessinsider.com/iot-smart-city-technology?r=US&IR=T. Accessed 27 Aug 2021.
10. Matthew Nitch S. The number of cars worldwide is set to double by 2040. 2018. https://www.weforum.org/agenda/2016/04/the-number-of-cars-worldwide-is-set-to-double-by-2040. Accessed 2018.
11. Milanes V, Villagra J, Godoy J, Simo J, Perez J, Onieva E. An intelligent V2I-based traffic management system. IEEE Trans Intell Transp Syst. 2012;13(1):49–58. https://doi.org/10.1109/tits.2011.2178839.
12. Liu, M., Yu, L., Guo, J., Guo, S., Guo, J., & Wen, H. (2007). Fuzzy logic-based urban traffic congestion evaluation models and applications. In: International Conference on Transportation Engineering 2007, pp. 1169–1174. https://doi.org/10.1061/40932(246)192
13. Transportation Research Board of the National Academies. Highway capacity manual 2010 (p. 18–6). Washington, D.C.; 2010.