Comparative Study between Traffic Control Methods Using Simulation Software

Abdulrahman Al-Kandari, Imad Al-Shaikhli, and Anas Najaa

***Abstract*—Traffic jams are a huge issue in urban streets. Intersections are one of the critical parts of the street that causes high amount of congestion if not managed properly. Solutions where made that revolve around fixed set of operations that attempt to find optimum time for each traffic light in the intersection. Several solutions were proposed over the last years that use a more intelligent and more dynamic than the previous ones. Some of those approaches, labeled as Intelligent Traffic Light Control System, are discussed in this paper. The methods are Webster, Dynamic Webster, Equal Interval and Optimum Equal. A simulation software will run those methods in a typical four-phase intersection and generate a report about their results. Each method is demonstrated and explained in the context of the simulation software. The methods are compared with emphasis on cycle interval time and flow rate. A conclusion is made about what can be improved in those methods.**

***Index Terms*—Isolated intersection, simulation, webster, Dynamic Webster (DW), Optimum Equal (OE), Equal Interval (EI).**

I. Introduction

Traffic congestion in urban cities is a major problem. It costs the individuals and the government time and money duo to wasted potentials and even higher probability of subsequent incidents specifically in developing nations where vehicular traffic is rapidly increasing [1]. Traffic Management Systems (TMS) thus implemented to solve this issue by optimizing the traffic flow to reduce traffic jam. Better approach has been discovered and improved during the last years called Intelligent Traffic Light Control System. Intelligent Traffic Light Control System can produce more optimal results through manual control of the road parameters [2]. Then the user sends those parameters to a microprocessor connected to the traffic lights in a certain intersection. Different setups where used to adapt to known road conditions like rush hours [3]. Those methods where rather fixed solutions that cannot adapt unless the user manually switches from one set of parameters to another. As a result, new adaptive control methods where created that continuously optimize the parameters depending on road conditions. One of those methods the well-known Webster's method. In his formula or algorithm, he used different concepts to calculate the optimum green time for each traffic light in an intersection. Another convenience of Intelligent Traffic Light Control System is the ability to simulate them using software easily with different options in this paper [4], [5], some of the previous methods are discussed including Webster, Equal Interval, Optimum Equal and Dynamic Webster in section two titled as methods explanation and flowcharts. The methods are converted in such a way that they can run on a simulation software of our design. This simulation software will simulate an intersection and the street conditions including specified traffic flow, cycle time, minimum green interval, method or algorithm in use and other parameters. The simulations in this paper lasted for 30 minutes for each algorithm. The software will generate a report after the session is completed that includes a lot of information that from we can conclude the efficiency of the used method. This discussion of the results is shown in details in section three titled as discussion and results with great focus on cycle interval and flow rate. Lastly, conclusion and future works related to the methods presented in section four.

Manuscript received July 30, 2013; revised October 5, 2013.

Abdulrahman Al-Kandari and Imad Al-Shaikhli are with the International Islamic University Malaysia, Computer Department (e-mail: aam_alkandary@yahoo.com, imadf@iium.edu.my).

Anas Najaa is with the Computer Education Department, Basic Education Collage, PAAET, Kuwait (e-mail: spidernet12@gmail.com).

II. Methods Explanation and Flowcharts

Traffic control uses several algorithms to manage the traffic flow on intersections. Those algorithms have strengths and weaknesses in different scenarios. In this section, each method is explained using a flowchart and mathematical representation of how the green time is distributed among the four roads in the intersection.

A. Equal Interval

This method is straight forward and the simplest method (See Fig. 1) of all the four methods currently mentioned. First, "Green Time" must be calculated by removing Red Interval and Yellow Interval from the Cycle Time as seen in (1):

$$WT = (TYI + TRI) \tag{1}$$

where *WT* is Wasted Time, *TYI* is Total Yellow Interval and *TRI* is Total Red Interval.

Then pure green interval is obtained then by subtracting the full cycle time given to the intersection from the time wasted on Red and Yellow Intervals, as seen in (2):

$$ACI = (CI - WT) \tag{2}$$

where *ACI* is Actual Cycle Interval, *CI* is Cycle Interval and *WT* is Wasted Time.

Then Green Time is distributed evenly among the four lanes by simple division, as seen in (3):

$$GI = \frac{CI - WT}{4} \tag{3}$$

DOI: 10.7763/IJMLC.2013.V3.353

where *GI* is Green Time and *CI* is Cycle Interval.

The flow chart for calculating Equal Interval shown in Fig. 1:

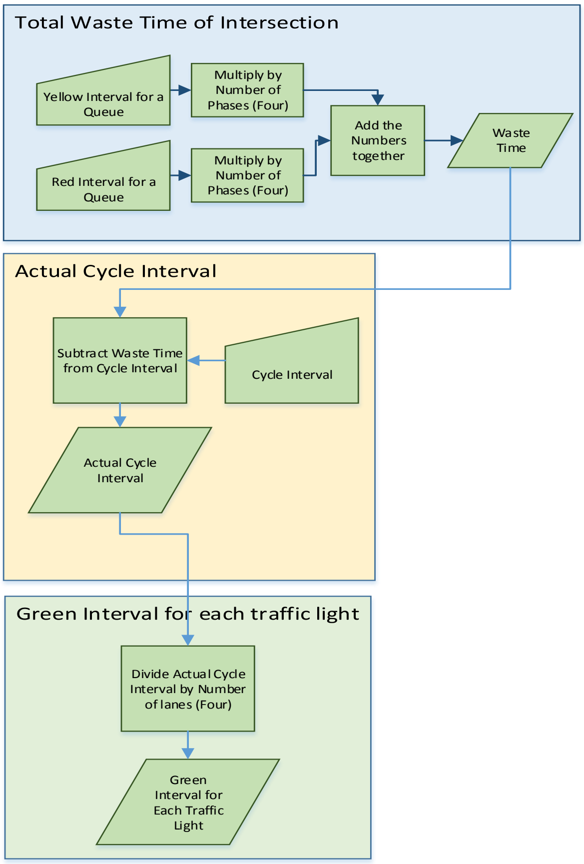

Fig. 1. Equal interval flow chart.

B. Optimum Equal

This method is just like Equal Intervals method; it gives each traffic light an equal Green Interval value. The difference resides in how this method calculates Actual Cycle Interval as seen in Fig. 2.

The equation multiplies the sum of the queues by the average cross ratio and subtracts the waste time to get the Actual Cycle Time as seen in (4):

$$ACI = ((q_1 + q_2 + q_3 + q_4) \times (\frac{r_1 + r_2 + r_3 + r_4}{4})) - WT \quad (4)$$

where (q) is the total queue of a road and (r) is the average cross ratio of the specified road.

What this means is instead of giving static Cycle Interval that does not scale with traffic load, the system decides to give Dynamic Cycle Interval that scales with the Queues as well as the average of the Cross Ratio.

Giving the final form of the equation 5:

$$GI = \frac{(\sum_{i=1}^{4} q_i) \times (\frac{1}{4}\sum_{i=1}^{4} r_i) - WT}{4} \quad (5)$$

where:

(q) is the queue of a road, (r) is the cross ratio of a road and(i) is the number of the road specified.

It is important to notice that by giving the ability to scale, an issue is exposed which can lead to the formula giving a very large number for Green Interval or sometimes a very small and unreasonable number. Therefore, minimum and maximum limits are tested against Actual Time before applying it.

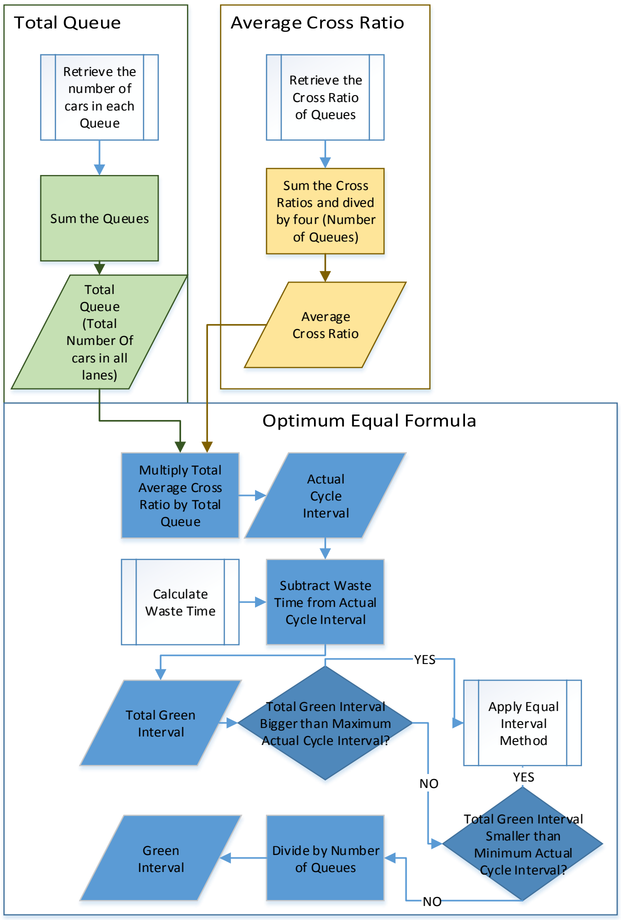

Fig. 2. Optimum equal flow chart.

The minimum value of Actual Cycle Time is obtained by summing all the Minimum Green Intervals of Lanes as seen in (6):

$$MACT = MGIQ_1 + MGIQ_2 + MGIQ_3 + MGIQ_4 \quad (6)$$

where *MACT* is Minimum Actual Cycle Time and *MGIQ* is Minimum Green Interval of Queue *X*.

Similarly, if the calculated value exceeds the specified Cycle Interval for this intersection then, the method will use instead the Equal Interval method described before as a solution. As a result, the Actual Cycle Time here will always be a variable number swinging between the Minimum Actual Cycle Interval and the Maximum Actual Cycle Interval.

C. Webster

Webster method calculates the actual cycle interval just like Equal Interval. However, it calculates the Green Interval depending on the current queue relative to the total queue number of all traffic lights. This is somehow similar to Optimum Equal but instead of looking at the intersection as a whole “Total Queue”, the method will focus on each Queue and the percentage of cars that exist in it and give it a suitable value out of the 100% of the Green Interval Time, see Fig. 3:

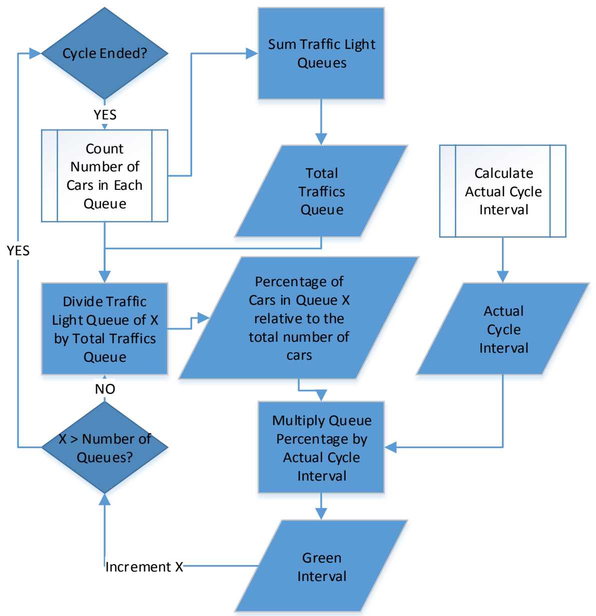

Fig. 3. Webster flow chart.

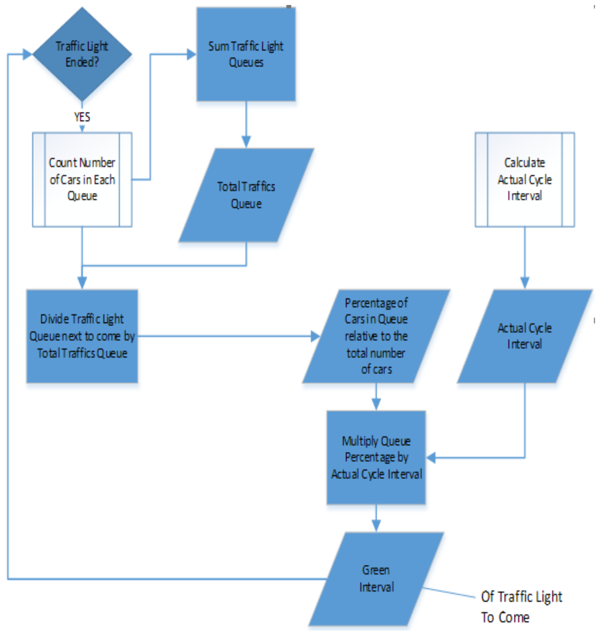

Fig. 4. Dynamic webster flow chart.

Equation 7 describes how the percentage of cars is calculated:

$$\frac{TLQ}{\sum_{i=1}^{4} q_i} = PCQ \qquad (7)$$

where q is the Total cars in a specified road, PCQ is Percentage of Cars in Queue and TLQ is Traffic Light Queue. Thus, the actual cycle interval will be divided according to these percentages as seen in (8):

$$GI = ACT \times \frac{TLQ}{\sum_{i=1}^{4} q_i} \qquad (8)$$

where q is the Total cars in a specified road.

Note that, this method executes once per cycle. Which means, the total queue is calculated only when the first traffic light in the cycle comes in the scene.

D. *Dynamic Webster*

This is identical to the Webster method except that the method here in the Green Interval calculation part does not wait the cycle completion in order to recalculate the total queue (as the Webster does) instead, it recalculates the total queue continuously and resets the Green Interval for each traffic light accordingly, see Fig. 4. The equation is identical to Webster as seen in (8).

III. Discussion and Results

The following section contains graphs that show a comparison between the different algorithms used to control the traffic flow. The iTraffic simulation software was used to run the simulations. Before discussing the results a brief review of the constants, decisive factors and other factors are discussed to provide a background about the simulation.

A. *Constants Used in the Simulation*

The simulations have fixed terms except for the Cycle Interval Time. Table I shows a list of the assumptions made before running the simulations:

TABLE I. The Fixed Variables in the Simulation

Option	Value
Number of Tracks	4
RTL Flow (Right To Left)	3600 c/h
LTR Flow (Right To Left)	7200 c/h
TTB Flow (Top To Bottom)	3600 c/h
BTT Flow (Bottom To Top)	1000 c/h
Yellow Interval	3.5 Seconds
Red Interval	1 Second
Minimum Green Interval	2.5 Seconds
U-Turns	Disabled
Left Turns	Disabled
Right Turns	Disabled
Default speed	60 km/h
Simulation Time	30 Minutes
Drawing Thread Sleeping Period	50 Millisecond

Track number is equivalent to lanes number. Currently the maximum number of tracks or lanes is four. U-Turns and the ability to turn left or right is excluded from this simulation to avoid anomalies in results and to focus more on the cross ratio than the nature of the road or the cars behavior. Drawing Thread is an Asynchronies method that fires up at predefined intervals to draw the cars on the screen. The interval or period set in this simulation is 50 Millisecond. The lower the period the smother the animation and the more recourses the software will need. Fifty Millisecond is the default value.

B. *Decisive Factor*

The methods included in the comparison and there short terms are:

1) E = Equal Interval
2) OE = Optimum Equal
3) W = Webster
4) DW = Dynamic Webster

Cycle Interval Times used in the comparison are:
60, 90, 120, 150, 180, 210, 240 Seconds

C. *Other Factors*

There are there factors that can affect the simulation results indirectly and contribute into the outcome. One of the factors is the machine performing the simulation. Some computers have better capabilities and performance in term of graphics and processing speed. Since the software uses DirectX to draw the cars on the screen the computer needs to be able to handle such load since the number of cars increases dramatically as the simulation runs for longer times. The CPU too plays a big role in the calculations and can affect the simulation results negatively by displaying anomalies in the data if the Processing speed is too slow.

Another factor is the zooming degree of the road. The farther you zoom away from the road the more cars the software is able to load into the lanes. Zooming out increases the storage of the lanes therefore making the simulation collect more data, which is better for analysis. Zooming in decreases the storage capacity of the lanes and reduces the data that can be collected. The only reason to actually zoom in if the simulation speed is affected or the computer cannot handle such load, then zooming in will be a reasonable solution to reduce the load and collect correct data.

Additionally, a random function adds cars to each road on random intervals. This provide a more realistic simulation of the road since adding cars directly and instantly to the road is not acceptable nor beneficial to the simulation. In addition, the simulator gives realistic speed to the cars, for example cars on the left most side of the road will move by the top speed set for the simulation but cars on the most right side of the road will move slower because the lane is considered a safe lane.

D. Charts Discussion

1) Line chart

The following section compares between the four methods in a single line chart with cycle times of 60, 90, 120, 150, 180, 210 and 240 seconds.

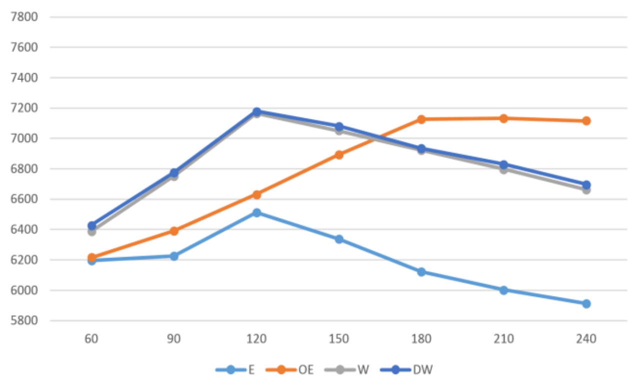

Fig. 5. Line chart that shows number of cars passed using Webster, Dynamic Webster, Equal Interval and Optimum Equal methods in term of cycle time or seconds.

The line chart in Fig. 5 displays the major difference between Equal interval, Optimum Equal, Webster and Dynamic Webster Algorithms. The X-axis shows the number of cars and the Y-axis shows the time of the cycles in seconds. The static cycle algorithms, namely Webster, Dynamic Webster and Equal interval, work almost at equal efficiency when the time is short. The starting point of Webster though is higher than that of Equal Interval duo to the fact that it distributes green time depending on the percentage of each queue relative to the total queue while Equal Interval simply distributes the time equally, in other words it lacks both dynamic distribution of time and dynamic adjustment of the cycle time itself. Note that When the time is short, managing cars becomes easy since the difference between times given to each road is minor and almost neglectable; you can only divide 60 seconds that much after all. The algorithms will not have much room to scale.

After the time increases, some of the algorithms will start to use the increased time to scale better than other algorithms. Those algorithms are namely Webster and Dynamic Webster. The scaling is good enough for 90 seconds. They can handle sudden burst in flow making them more reliable for real life scenarios. That being said, equal interval doesn't seem to perform quite as well as the other algorithms, in fact it falls short as it used to perform better with 60 seconds cycle time. This is duo to the algorithms dividing time in an inefficient matter. As seen in Table I the number of cars in each road is different, some of them has 7200 cars flowing each hour, some have 3600 cars flowing each hour and the BTT road has 1000 cars flowing each hour. Since the simulation ran for about 30 minutes, the theoretical number of cars that should flow in each road is half of the amount that should flow each hour. However, in real life scenario this number is decreased for several factors that include driver's behavior, time of the day and over all road conditions. Therefore, the number of flowing cars each 3 minutes is something close to the theoretical number. That being said the reason why equal interval falls short is obvious, for there is different number of cars that flow in each lane. Equal interval will give each road the same amount of time and that is not an efficient solution when there are different traffic flows for each road.

When the time increases to 120 seconds, the algorithms start behaving differently. Equal interval has already fell short before but it rises again to a degree close to Optimum Equal because almost all the roads have green time close to their optimal green time, which is ought to happen if you keep increasing the cycle time. Sadly, this effect does not last because later on the time wasted on roads with little traffic flow will be greater than the time saved on the roads with huge traffic flow as seen in the decline of Equal Interval upon increasing the cycle time. Webster and Dynamic Webster behave similarly as they reach the highest spot at 120 seconds cycle time and then decline as they cannot scale the cycle time itself. Even though there is a percentage scaling, the percentage of cars does not translate that well to how many cars actually pass, as the number of cars increases. Optimum equal is able to rise unlike other algorithms, which is due to its dynamic cycle scaling to the number of cars on the roads. Dynamic cycle means the algorithm can pick a cycle time from a range that extends from minimal green time to the maximum cycle time allowed. Optimum equal stops increasing at 180 seconds mark and remains steady to 240 seconds mark. While dynamic Webster and Webster decline at a steady rate from 120 seconds to 240 seconds. In addition, from there on as the number of cars increases the scaling falls short for every algorithm without a dynamic cycle time.

2) Column chart

The following section compares the four methods separately, times used are 60, 90, 120 seconds cycle interval.

As seen in Fig. 6, dynamic Webster outperform all other algorithms at minimum cycle interval of 60 seconds. Followed by Webster. Note that the difference between DW, W and OE, E is not huge, only ranging from 200 to 250 cars duo to the small cycle interval.

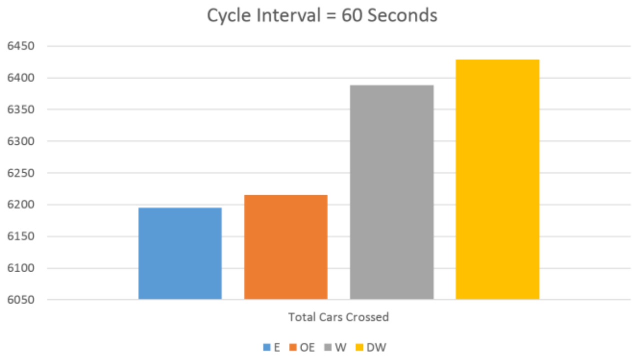

Fig. 6. Column Chart that compares different methods at 60 seconds cycle time.

As seen in Fig. 7, Optimum Equal improves a little bit than in 60 seconds cycle time. This improvement keeps scaling as the cycle time increases, unlike Equal Interval. Also Webster and Dynamic Webster increases steadily by a range of 100 to 300 cars.

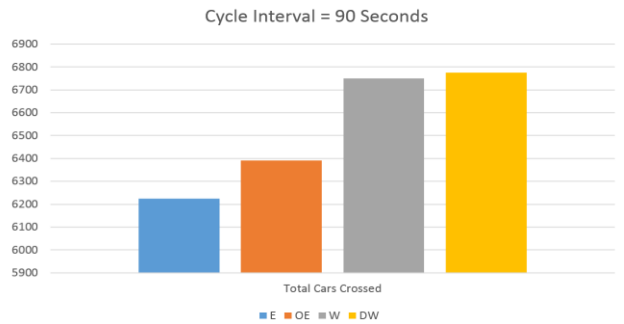

Fig. 7. Column chart that compares different methods at 90 seconds cycle time.

In Fig. 8, Optimum Equal keeps rising, as it scales better relatively to other methods, which their flow rate is increased by a small amount. At the same time, Equal interval reaches its peak. This means that Equal Interval can no longer get better if the cycle time increases.

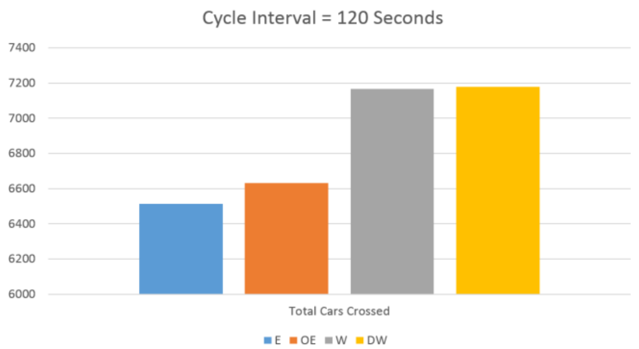

Fig. 8. Column chart that compares different methods at 90 seconds cycle time.

IV. Conclusion and Future Work

After running the previous simulations, we can conclude that dynamic adjustment of cycle time and green time distribution for each road is the key factor in controlling traffic flow effectively to reach optimum solution for each intersection. In this paper, Dynamic Webster and Equal Interval played a huge role in optimizing traffic flow in different ways by refreshing queue status each time in Dynamic Webster and scaling cycle time in Optimum Equal. Those two methods should be the focus of future work as they can be further improved by adjusting their algorithms to avoid there weakness.

Appendix

Software used in the simulation is displayed in Fig. 9:

Fig. 9. Software built using VB.NET and SQL.

Sample of the report generated by the simulator is displayed in Fig. 10:

Properties	RTL	LTR	TTB	BTT
Method	Equal	Equal	Equal	Equal
Projected Flow - c/h	1,000	1,000	1,000	1,000
Tracks Count	4	4	4	4
Simulation Time	00:00:00	00:00:00	00:00:00	00:00:00
Specified Cycle Interval	00:01:00	00:01:00	00:01:00	00:01:00
Minimum Green	00:00:03.50000	00:00:03.50000	00:00:03.50000	00:00:03.50000
Red Interval	00:00:00.50000	00:00:00.50000	00:00:00.50000	00:00:00.50000
Yellow Interval	00:00:03.50000	00:00:03.50000	00:00:03.50000	00:00:03.50000
Waste Time	00:00:04	00:00:04	00:00:04	00:00:04
Actual Cycle Interval	00:00:00	00:00:00	00:00:00	00:00:00
Cycles	0	0	0	0
Total Green Time	00:00:00	00:00:00	00:00:00	00:00:00
Total Green Time %	NaN	NaN	NaN	NaN
Car Size	34x17	34x17	34x17	34x17
Speed Range	15-60	15-60	15-60	15-60
Storage Length	678	679	377	377
Queue Weight	56	56	32	32
Crossed ahead	0	0	0	0
Crossed ahead %	NaN	NaN	NaN	NaN
Right Turns	0	0	0	0
Right Turns %	NaN	NaN	NaN	NaN
Left Turns	0	0	0	0
Left Turns %	NaN	NaN	NaN	NaN
U-Turns	0	0	0	0
U-Turns %	NaN	NaN	NaN	NaN
Total Cars Crossed	0	0	0	0
Cross Ratio	0.00	0.00	0.00	0.00
Actual Flow - c/h	0	0	0	0
Queue Left	0	0	0	0
Traffic State	Off	Off	Off	Off

Fig. 10. Report Sample generated by the Software.

Some of the options the user can control are displayed in Fig. 11, Fig. 12, Fig. 13 and Fig. 14:

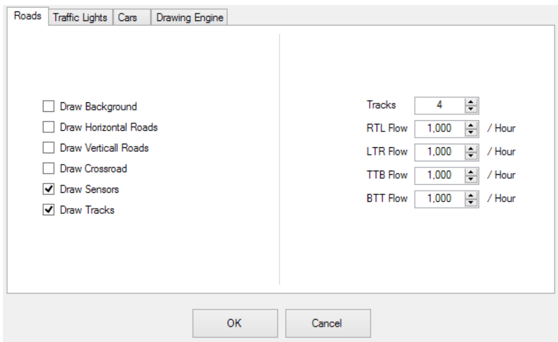

Fig. 11. Options Related to the roads.

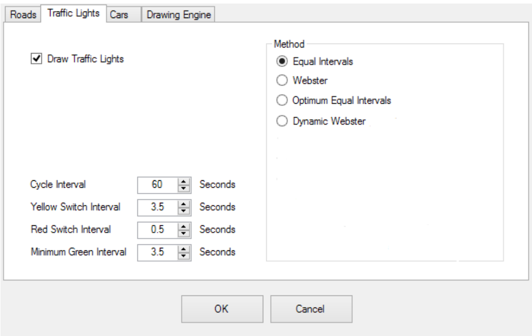

Fig. 12. Options related to the traffic lights.

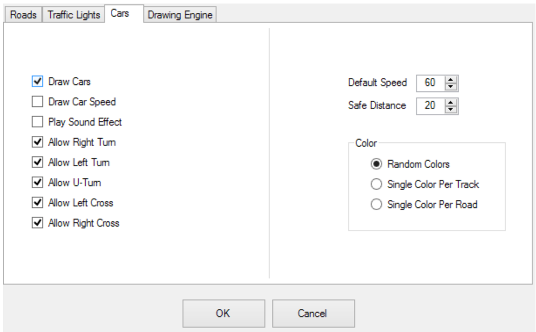

Fig. 13. Options related to the cars.

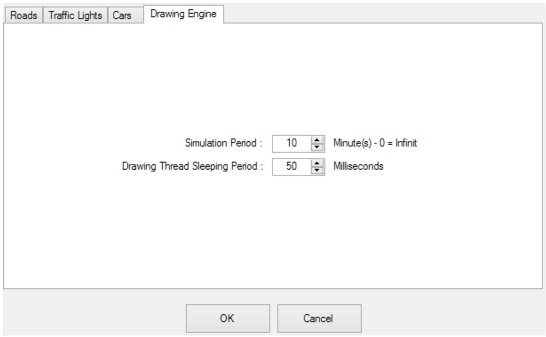

Fig. 14. Options related to the drawing engine.

REFERENCES

[1] K. Jraiw, "Urban road transport in Asia's developing countries: Safety and efficiency strategy," *Transportation Research Record*, vol. 1846, pp. 19-25, 2003.

[2] R. L. Gordon, W. Tighe, *Traffic Control System Handbook*. Washington, DC: FHWA-HOP-06-006, U.S. DOT, FHWA, 2005.

[3] F. V. Webster, *Traffic Signal Settings*, Road Research Technical Paper No. 39, Road Research Laboratory, Her Majesty's Stationery Office, London, U.K., 1958

[4] C. Bielefeldt, D. Bretherton, and C. Toomey, "The role of online urban signal control in congestion and incident detection and management," presented at the IEE Colloquium on Incident Detection and Management (Digest No: 1997/123), 1997.

[5] O. U. Chinyere, O. O. Francisca, and O. E. Amano, "Design and simulation of an intelligent traffic control system," *International Journal of Advances in Engineering & Technology*, vol. 1, pp. 47-57, 1963.

A. Alkandari was born on January 29, 1981 in Kuwait. A. Alkandari finished his bachelor degree of Computer Engineering in Kuwait University, Kuwait in 2004, and the master degree of Computer Engineering in Kuwait University, Kuwait in 2011. A. Alkandari studied Ph.D. of Computer Science in International Islamic University Malaysia, Malaysia starting from 2011 to 2014.

He worked on Civil Service Commission, Centre of Information Systems from February 2005 to end of May2006 as a project coordinator for" Integrated System"(e-Government), then Mobile Telecommunication, Kuwait (MTC) as a support engineer from June 3, 2006 to March 4, 2007. After that, He became a faculty member at the Public Authority for Applied Education & Training (Telecommunication & Navigation Institute) on computer department from March 4, 2007 to September 16, 2008. Finally, He is a faculty member at the Public Authority for Applied Education & Training (Basic Education Collage) on computer department from September 16 2008 till today. He is interest in intelligent systems, traffic engineering, algorithms, smart cities and wireless sensor network.

Mr. Alkandari is a member of Kuwait Society of Engineers and a member of Kuwait Information Technology Society.

I. Alshaikhli was born on March 26, 1962 in Iraq. I. Alshaikhli finished his bachelor degree of Maths in Iraq in 1985, master degree of Computer Science in Iraq in 1991, and the Ph.D. of Computer Science in India in 2000.

He worked as a system analyst since 1985, then the head of the dept. of Computer Information System at Al-Rafidain University from 2003 to 2005. After that, He was a faculty member and dean of College of Computer Eng. and Science, Gulf University- Kingdom of Bahrain from 2006 to 2010. Finally, He is a Faculty Member of Computer Science, International Islamic University Malaysia from 2010 till today. He is interest in cryptography, steganography, genetic algorithms, neural network and biometrics.

I. Alshaikhli is the editor in chief of JACSTR (Journal of Advanced Computer Science and Technology Research). He is also a member of Malaysian Society for Cryptology Research – 2012, the General chair - ACSAT 2012, a member of the Board of Governors/ DSR-2012. Associate editor: Pakistan Journal of Engineering, Technology and Science (PJETS), 2011. He is an IEEE member in Computational Intelligence. Committee member, PACLING 2011 Malaysia, a member of Iraqi Computer Society since 1992. He is also a technical program committee for ISCI 2012.

A. Najaa was born on February 12, 1991 in Kuwait. A. Najaa finished his bachelor degree of Computer Education in Public Authority for Applied Education and Training, Kuwait on 2013.

He worked as a teacher assistant in Eyass Bin Moath Secondary school. He was one of the authors who published a paper in IEEEXplore "SVG and Arabic font challenges and improvements". Alahmad, M. A., Al-shaikhli, I., and Najaa, A. M., in *Proc. Information and Communication Technology for the Muslim World (ICT4M), 2013 5th International Conference*. He is interest in security, programming, designing and robotics.

A. Najaa was the vice president of Computer Club for 1 year, *PAAET* and a Member of Google Developers Group/Kuwait.