Control of Traffic Light Using Fuzzy Logic and Harmony Search Algorithm

Erwin Sitompul
Study Program Electrical Engineering
Faculty of Engineering, President University
Bekasi, Indonesia
e-mail: sitompul@president.ac.id

Agustinus Bunawan
Study Program Electrical Engineering
Faculty of Engineering, President University
Bekasi, Indonesia
e-mail:agustinus_bun@ymail.com

Abstract—**Traffic light controller holds an important role in maximizing traffic flow. However, normal traffic light controller with fixed phase duration is ineffective to handle traffic with variable density. Therefore, this paper proposes a fuzzy logic controller to improve the flexibility of traffic light controller in handling variable traffic density. Furthermore, the fuzzy logic controller is optimized using Harmony Search algorithm. The proposed fuzzy logic controller is marked by its capability to decide which lane will have green phase. Also, it is capable to extend and terminate a green phase or a red phase. An observation on an intersection at Bekasi, the fifth largest city in Indonesia, is done to provide a real case simulation for comparing the proposed fuzzy logic controller and the existing controller. The simulation shows that the optimized fuzzy logic controller outperforms the existing fixed-duration controller.**

Keywords-traffic light controller, fuzzy logic; Harmony Search algorithm

I. Introduction

The increasing number of vehicles in Indonesia has led to traffic congestion problem especially in major cities. Travel time, environment quality, and road safety are adversely affected as the result of traffic congestion. It may be caused by a number of factors such as density of vehicles on the road, driving habits, social behavior, and traffic light system. The traffic light system is in turn highly influenced by the control scheme applied into it. In Indonesia, the traffic light controller commonly implemented is a fixed-duration one. This kind of traffic light controller allocates a fixed time for greed phase and red phase, which is rather ineffective during the time interval in which the traffic density is low. The fixed-duration traffic light controller keeps the programmed phase duration in a lane even when there are no vehicles coming to it. Therefore, a new traffic light controller based on fuzzy logic is designed to improve the fixed-duration traffic light controller. Furthermore, the Harmony Search algorithm is employed on the purpose of optimizing the membership function of the fuzzy logic controller.

This paper has been organized as follows: first a brief explanation about the proposed fuzzy logic controller will be presented. Afterwards, the lane modeling is discussed. The application of Harmony Search algorithm to optimize the fuzzy membership function follows in the next section. Finally, the simulation results using artificial data and observation data are presented.

II. Fuzzy Logic Controller

In traffic light control using fuzzy logic, there are several approaches introduced by researchers. One considers the number of cars to decide the extension time for green phase [1]. Sensors are placed behind the traffic light and at a certain distance away from the traffic light. The number of cars coming during a green phase is named *arrival* while during a red phase denoted as *queue*. The extension of green phase is decided based on *arrival* and *queue*.

Another approach uses multi-level signal control [2]. The first level is the fuzzy green phase extender function and the second level acts as the fuzzy phase selector. The fuzzy green phase extender decides whether to extend or to terminate the current green phase. If the decision is to terminate the green phase, the lane which will be given the next green is decided by the fuzzy phase selector, based on the average queuing length of all lanes with red phase.

The new fuzzy logic controller proposed in this paper has two main parts. The first part is the **Urgency Module**, which processes the inputs from each lane to become the urgency level using fuzzy logic. The second part is the **Decision Module**, which decides whether to extend or terminate the current green phase and give it to another lane, based on the urgency level of each lane. The general overview of the proposed fuzzy logic controller can be seen in Fig. 1.

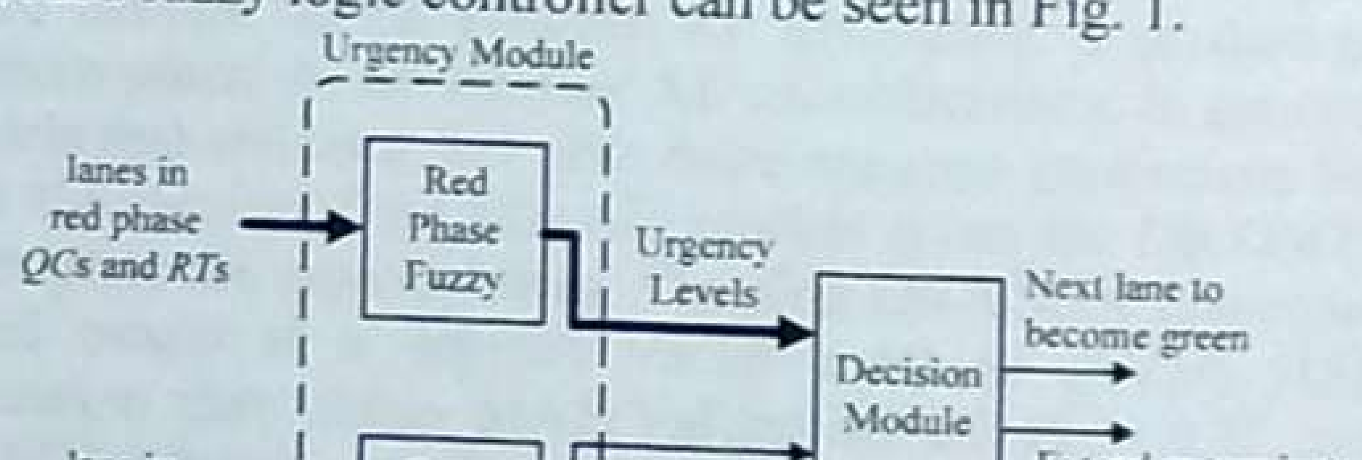

Figure 1. General scheme of the proposed fuzzy logic traffic light controller

The inputs for the **Urgency Module** are the number of queuing cars (*QC*) and the duration of the current traffic light signal from each lane. The number of queuing cars of each lane contributes with identical weight to the urgency level of the lane. The increasing number of queuing cars increases the urgency level of the particular lane. For lanes currently in red phase, the increasing duration of traffic light signal

978-1-4244-9252-7/11/$26.00 ©2011 IEEE

increases the urgency level, while it will decrease the urgency level for the lane currently in green phase. Therefore, there are two fuzzy logics employed in the urgency module, **RedPhase Fuzzy** and **GreenPhase Fuzzy**.

The **RedPhase Fuzzy** processes the urgency for all lanes currently in red phase to become the urgency level of each of the lanes. The **GreenPhase Fuzzy** evaluates the inputs from the lane currently in green phase. After evaluating each lane's urgency level, the **Decision Module** decides which lane will be given a green phase for the next turn. If the lane selected is the lane currently in green phase, this means extending the green phase for 1 second. On the other hand, if the selected lane is the lane currently in red phase, the decision means giving the green phase to the respective lane and holds it for the next 15 seconds.

The **RedPhase Fuzzy** has two inputs and one output. They are Queuing Car Number (*QC*) and Red Time (*RT*) of all red lanes. The **GreenPhase Fuzzy** also has two inputs, Queuing Car Number (*QC*) and Green Time (*GT*) of the lanes currently in green. The output for both the **RedPhase Fuzzy** and **GreenPhase Fuzzy** is the urgency level (*UR*).

For **RedPhase Fuzzy**, *QC* refers to the numbers of cars remaining on a certain lane during a red phase. The lane with higher *QC* value has higher urgency level. A certain lane might keep getting the highest urgency level when the traffic on that lane is high. Therefore, a second input *RT* is needed, which refers to the duration of red phase. The urgency level of a lane will increase as the *RT* increases.

In **GreenPhase Fuzzy**, *QC* refers to the numbers of car that remain on a lane during a green phase. The second parameter is *GT*, which refers to the duration of green phase. In opposite to *RT*, the increment of *GT* will result in the decrement of urgency level of the particular lane.

In the proposed Fuzzy Logic controller, each fuzzy variable are composed of five linguistic variables which are: Zero (Z), Small (S), Medium (M), Large (L), and Huge (H). Their initial membership functions arrangement follows the arrangement shown in Fig. 2.

The **Decision Module** conducts the decision making of which lane will be given the next green phase. The controller checks the lane condition every second. If the lane decided to be the next green is currently in red phase, the controller terminates the present green phase, gives it to the chosen lane, and hold the green phase for at least 15 seconds before the cycle of checking the urgency level is repeated.

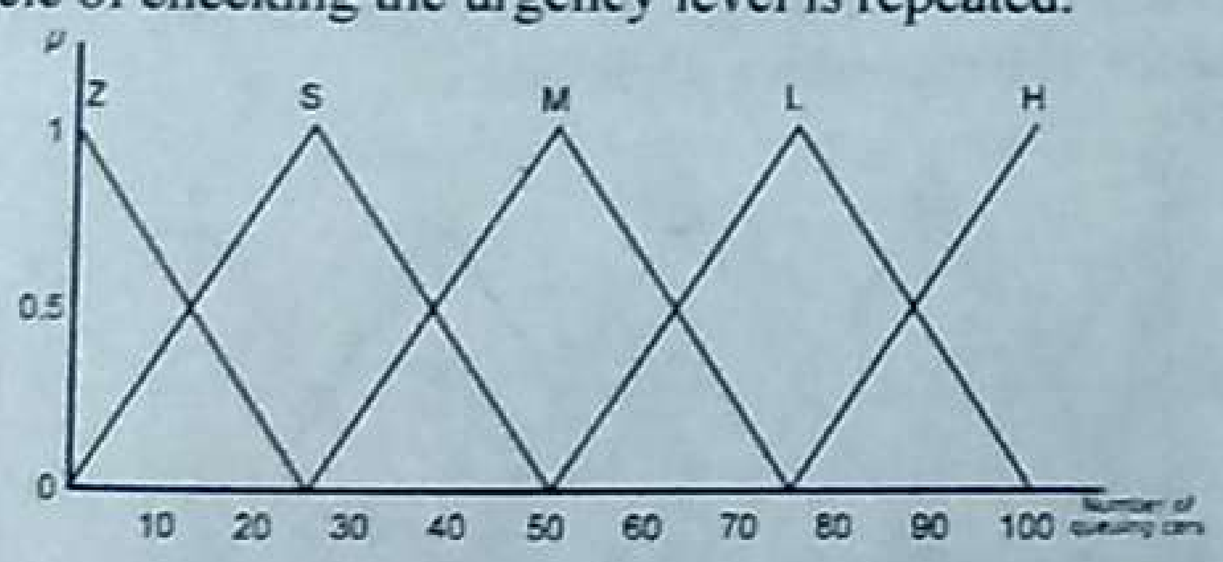

Figure 2. The initial arrangement of the fuzzy membership functions

The next step is to set the rule of the fuzzy logic controller. A total of 20 linguistic rules are developed for both **RedPhase Fuzzy** and **GreenPhase Fuzzy**, as shown in Table 1. and Table 2.

In some case, it is possible to encounter more than one lane in red phase having the highest urgency level. When occurs, the **Decision Module** will choose the nearest turn based on a predetermined cycle after the current green lane. For example, consider a 4-way intersection: A, B, C, and D. The cycle of green phase is set to be A - B - C - D, and back to A, while the lane that currently holds green phase is C. Assume that the urgency levels of both B and D are equal and also higher compared to the other two, A and C. Then, decision module will gives D the next green phase based on the given cycle.

TABLE I. RULES OF THE REDPHASE FUZZY

Rule	Inputs		Output	Weight
	QC	*RT*	*UR*	
1	Z	none	Z	1
2	S	none	S	1
3	M	none	M	1
4	L	none	L	1
5	H	none	H	1
6	none	Z	Z	1
7	none	S	S	1
8	none	M	M	1
9	none	L	L	1
10	none	H	H	1

TABLE II. RULES OF THE GREENPHASE FUZZY

Rule	Inputs		Output	Weight
	QC	*GT*	*UR*	
1	Z	none	Z	1
2	S	none	S	1
3	M	none	M	1
4	L	none	L	1
5	H	none	H	1
6	none	Z	H	1
7	none	S	L	1
8	none	M	M	1
9	none	L	S	1
10	none	H	Z	1

III. LANE MODEL

In order to show the effectiveness of the proposed fuzzy logic controller, a lane model is developed. A venture to exactly model a real lane is actually very complex. There are many factors that affect the lane condition, such as weather condition, driver habit, and car dimension.

Considering the complexity possessed by the real lane, several simplifying assumptions are taken:

1. All cars queue in order.
2. The speed of cars is constant.
3. The dimension of cars is identical.
4. The traffic light signal will be only red and green.
5. No motorcycles are on the road.
6. All car drivers have the same driving behavior.

The developed lane model is represented by a group of cells that are arranged in rows and columns. Fig. 3 shows a lane with 4 rows and 10 columns, which is used in this paper. One cell is designed only to be only occupied by one car at a time. Therefore, the lane capacity is 40 cars.

Every car coming to the lane will occupy the last columns first. The next second, it will move forward for one column. If the next column is fully occupied, the car will stay on the current column.

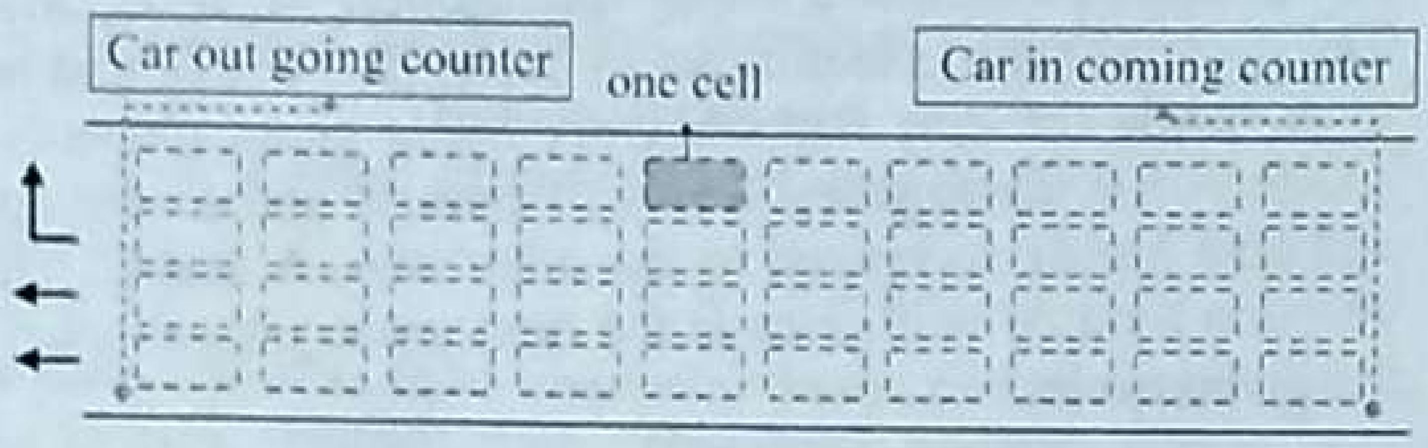

Figure 3. Lane model

The concept for the lane model during a green phase can be explained as follows: when a green phase is issued, the cars will start moving from a stop position. In order to emulate the reality, the movement of cars in green phase follows several rules as follows:

1. If a green phase issued at $t = 0$, then there will be no car going out in the first 5 seconds, until $t = 5$ s.
2. For the next 3 seconds after rule 1 is applied, the car will move forward through the lane with the velocity of 2/3 cell per second.
3. For rest duration of green time, the car will move with the velocity of 1 cell per second.
4. If the next column is fully occupied, the car will not move and wait for 1 second.

IV. Harmony Search Algorithm

The performance of a fuzzy logic controller depends on factors such as fuzzy rules, fuzzy membership functions, and fuzzy logical operators. In order to obtain the most suitable fuzzy membership functions for the Fuzzy Logic controller proposed in this paper, they are optimized using the Harmony Search Algorithm. The overall optimization process is shown in Fig. 4.

The **Harmony Search (HS)** algorithm has 5 steps [3].

Step 1 Initialization of optimization parameters: In this paper, the objective function $f(\mathbf{x})$ is the average waiting time where $\mathbf{x}$ is the vector of fuzzy membership function parameters. Each fuzzy variable has three parameters, as shown in Fig. 5 for *RedTime*. The three parameters are adequate to define the five membership functions. Since the number of all fuzzy variables is 6, then the total number of parameters to be optimized is 18.

After setting the objective function, the parameters of HS algorithm are defined. The values of HS parameters chosen are $HMS = 5$, $HMCR = 0.7$, $PAR = 0.45$, and $Bw = 5$.

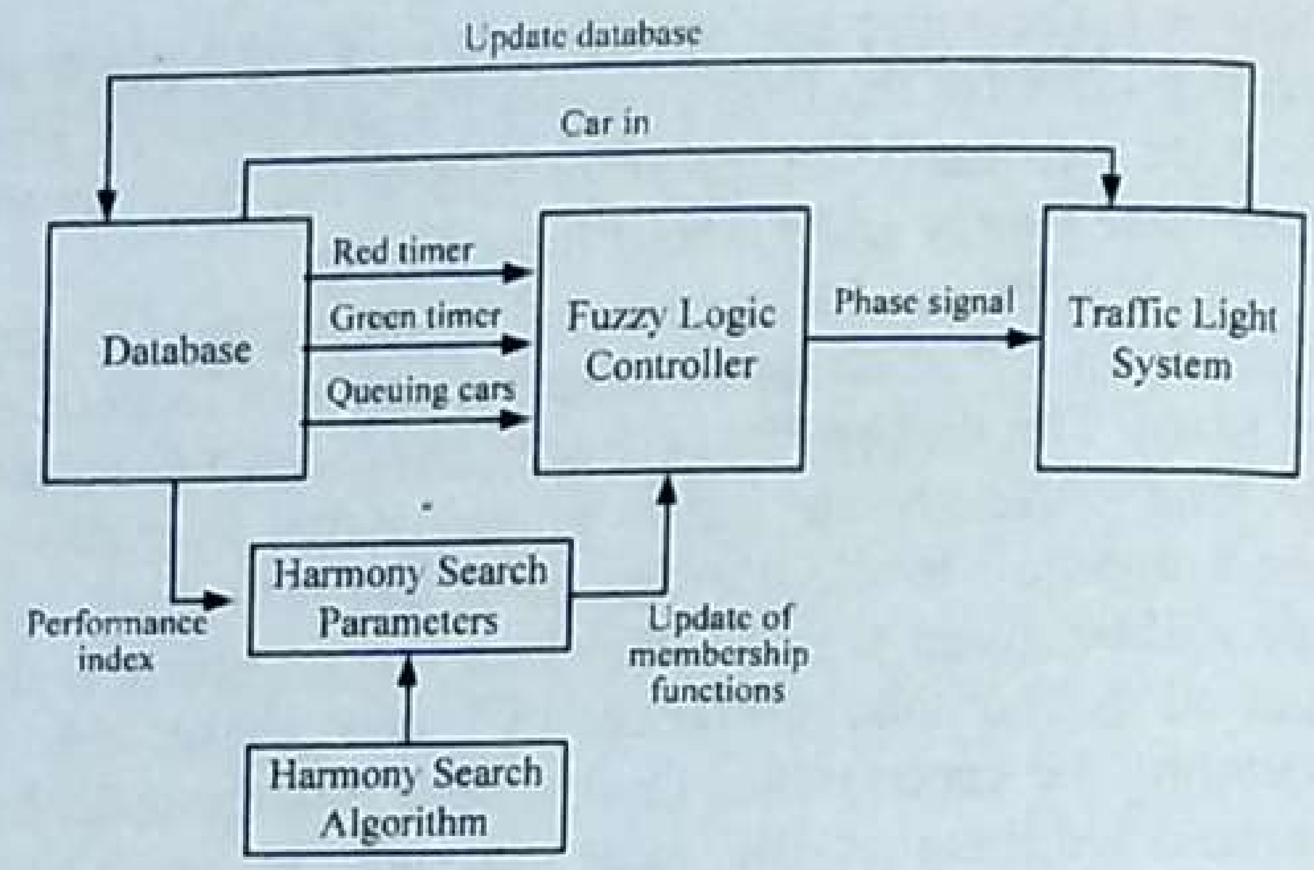

Figure 4. The overall optimization process

Step 2 Initialization of harmony memory (HM): In this step, HS generates a set of solution vectors as the initial HM. Each solution vector is sorted based on a certain criterion. The criterion is the value of the first parameter of a fuzzy variable must be less than the second and the second must be less than the third. Then, the HM is sorted based on the result given by the objective function.

Step 3 Improvise a new solution vector: In this step, **HS** generates a new solution vector based on the three rules: memory considerations, pitch adjustments, and randomization. The possibility that x_i will be chosen from *HM* is determined by *HMCR* while (1–*HMCR*) is the possibility of randomly choosing the solution from feasible candidates. If the selected solution is from the memory, it will be pitch-adjusted with probability of *PAR*. The boundary for every parameter is [0,100].

Step 4 Update the *HM*: In the beginning of this step, the generated new solution vector is transferred to the variable of the objective function. Then, the objective function processes the solution vector and returns the average waiting time of car. It is then compared with the worst result stored in the *HM*. If the new result is better than the worst stored result, then the new solution replaces the worst solution. After that, *HM* is sorted again based on the objective function results.

Step 5 Repeat step 3 and 4 until the stopping criteria are reached: In this step, HS checks the stopping criteria. If the improvement of the objective function is saturated for a certain iteration cycles, then the search algorithm will be stopped. In this paper, the HS algorithm will stop if the improvement of the objective function is less than 0.01 second for 300 successive iterations.

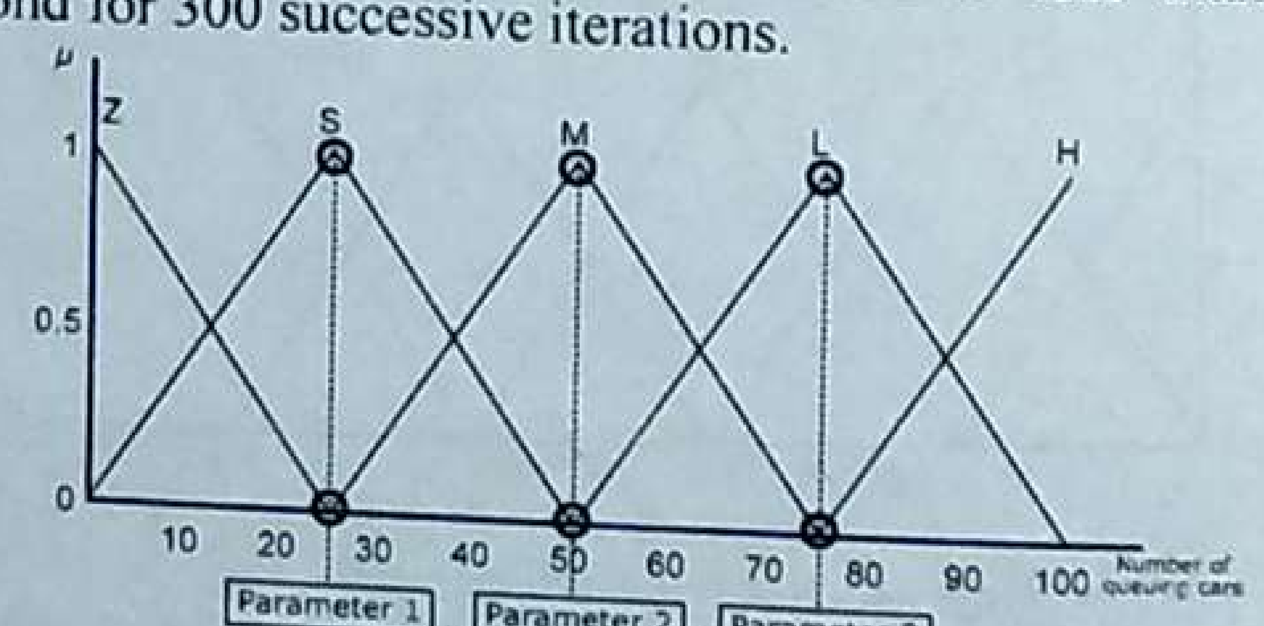

Figure 5. Parameters of fuzzy variable *RedTime* in RedPhase Fuzzy

TABLE III. RESULT OF HS ALGORITHM FOR SOME ITERATION NUMBERS DURING A TEST WITH ARTIFICIAL DATA

Iteration number	HM	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	$f(x)$
Initial value	1	25	67	80	1	45	56	7	28	90	48	92	98	56	65	77	0	11	54	34.222
	2	40	41	51	17	52	64	2	80	84	8	46	70	19	45	82	1	31	88	35.462
	3	1	20	45	2	62	79	76	89	91	33	38	76	50	56	77	48	78	80	35.948
	4	33	84	88	48	56	62	62	66	69	51	52	71	61	82	97	13	32	59	38.017
	5	20	47	58	67	68	94	74	77	87	50	63	99	45	52	79	13	17	22	52.693
100	1	24.7	46.5	64.1	64.6	74.5	99	10.3	36.7	65.1	71.2	71.2	85.6	61.8	92.4	94.7	28.4	83.9	91.1	26.247
	2	18.7	20.4	46	49.1	66.7	73.3	1	10.5	48.9	65.1	71	87.9	51.8	82	92.1	3.2	12.3	77	26.267
	3	20.1	45.9	66.7	78.3	83.9	93.8	0.5	14.7	36.7	48.9	62	87.5	50	85	88.6	33.1	45.4	77.6	26.330
	4	50.9	71.4	74.3	78.3	78.3	87.9	8.2	28.1	35.7	62.6	71.2	87.5	60.3	77.1	77.8	12.3	27.3	77.6	26.765
	5	45.9	62.9	65.7	66.7	66.8	73.5	0.5	14.9	31.5	48.9	75.3	87.9	78.5	85	99.7	12.3	39.9	49.8	26.821
562	1	45.9	61.4	62.5	68.1	79.7	99	5.9	10.3	44.6	52.3	71.2	88.8	73.3	91.7	94.7	28.4	29.7	78.3	25.457
	2	31.9	47.8	59.3	61.3	77.6	95.5	0.5	7.9	48.9	58.4	58.9	85.5	49.2	74.1	94.7	23.7	41.3	98.7	25.505
	3	28.5	46.3	53.6	54.7	65	65.4	0.5	1.1	43.5	63.5	87.3	90.1	38.2	73.3	99.1	23.7	36.7	91.5	25.511
	4	45.9	50.8	62.5	74	76	98.2	3.6	7.9	46.4	58.4	74.1	99.1	26.7	89.7	100	30.2	34.2	73.8	25.552
	5	33.2	59.3	65.4	70	75.9	98.4	0.4	0.5	44.6	71.2	79.3	85.5	73.3	74.1	100	28.4	36.7	98.7	25.553

V. RESULTS AND DISCUSSIONS

A. Controller Feature

The graphs in Fig. 6 show the information of green phase cycle and the number of queuing cars. The upper graph shows the lanes allocated with green cycle, "1" for north, "2" for east, "3" for south, and "4" for west. The lower graph shows the number of queuing cars on each lane, thin dotted-line for north, thin solid line for east, thick solid line for south, and thick dotted line for west. It can be seen that the cycle of the green phase is flexible to the lane condition. As an example, for the time between $t = 100$ s and $t = 200$ s, the cycle does not follow a certain repetitive order. This occurs because the fuzzy logic controller makes decision based on the lane conditions.

Another feature of the controller is that it will give a lane a green phase regardless the number of queuing cars, if the lane is already in red phase for a long duration. West lane already got red phase since $t = 116$ s. Although the number of queuing cars on it is significantly smaller compared to the other lanes, it is given a green phase at $t = 202$ s. The decision is based on the magnitude of red time. This guarantees that no lane will go red all the time.

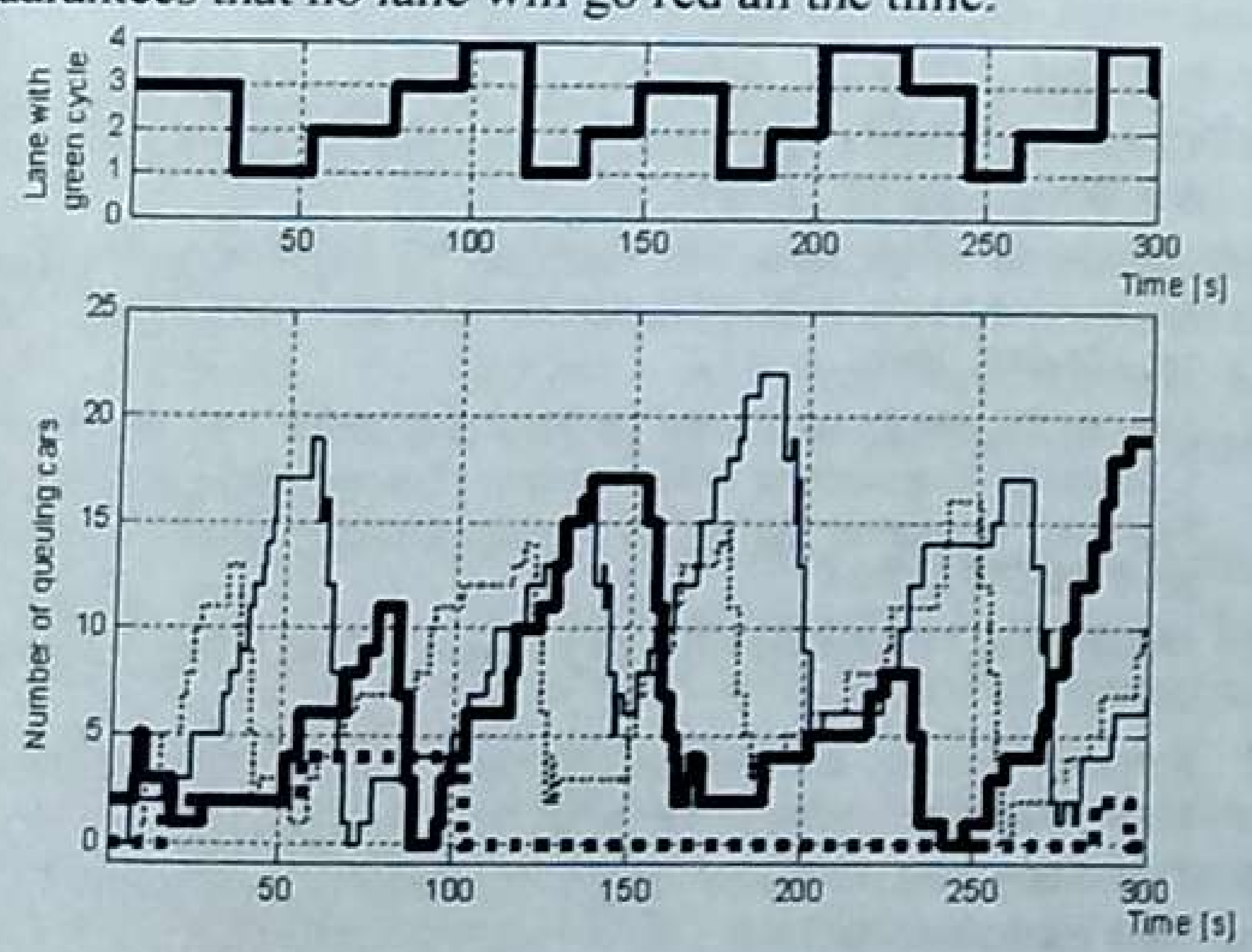

Figure 6. Test result of fuzzy logic traffic light controller with artificial traffic flow

B. HS Algorithm Test

The HS algorithm is devised to optimize the membership function of the fuzzy logic controller. The chosen objective function is the average waiting time of each car. A lane model with artificial car density is developed to demonstrate the ability of this algorithm, with the data comprised in CI_1.

Table 3 shows the progress of an optimization. *HM* is initially filled with randomly generated solution vectors within the aforementioned bounds. The vectors in *HM* are sorted according to the values of the objective function that they produce. Next, a new harmony vector is generated based on three rules. Their probabilities are distributed as follows: memory considerations with 38.5% ($0.7\times0.55 = 0.385$), pitch adjustment with 31.5% ($0.7\times0.45 = 0.315$) and randomization with 30% ($1-0.7 = 0.3$).

As the iteration continues, the solution becomes better. After 562 iterations, the improvement of the best memory of *HM* is less than 0.01 second for 300 successive iterations and the stopping criteria are reached. The best solution vector in *HM* is then considered as the optimized solution.

The optimized membership function of the fuzzy logic controller is validated using completely new sets of data, since the optimized fuzzy logic controller must also perform well in dealing with unseen data. Table 4 shows the validation results. The data in column CI_1 is the result using the known optimization data, while CI_2, CI_3, and CI_4 using unseen data. The thick frames mark the lowest average waiting time for each data set. It can be concluded that the optimized parameters also perform well with unseen data.

TABLE IV. AVERAGE WAITING TIME (IN SECONDS) FOR DIFFERENT SETS OF DATA USING OPTIMIZED MEMBERSHIP FUNCTIONS

Iteration number	CI_1	CI_2	CI_3	CI_4
203	26.247	24.711	24.906	26.543
214	25.945	26.178	25.288	25.922
225	25.851	24.060	23.850	25.507
261	25.736	26.613	26.909	25.586
562	25.457	23.703	23.893	24.908

C. Case Study

A case study was conducted on a 3-way intersection in Bekasi, the fifth largest city in Indonesia, as shown in Fig. 7. The existing traffic light controller at the intersection is a fixed-duration one. During a low traffic density, this fixed-duration type is not effective in controlling the traffic flow. This condition is observed at around 13:00 – 14:30 during a Tuesday, a Wednesday, and a Thursday.

The leftmost lane of each road is an always-can-go type. Thus, the data collection for *CI* will not consider the cars passing through that lane. The data from three observation days are denoted as CI_{1S}, CIS_{2S}, and CI_{3S}, where CI_{1S} are used in the optimization. Each ***CI*** data consists of the data of incoming cars from the three ways. The observation on each day was done for 1 hour.

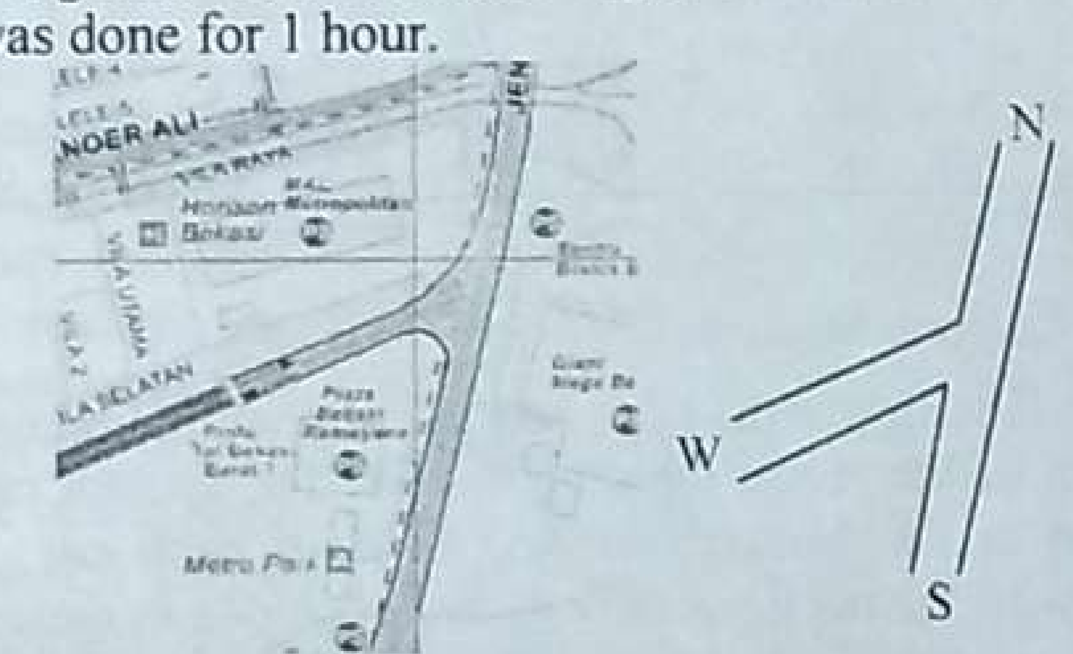

Figure 7. A 3-way intersection in Bekasi, Indonesia

1) Lane Model. Here the existing lane model is tuned to reflect the real traffic flow on the intersection. Data from five consecutive cycle is used. First, when the traffic light turns green, the time needed for the initially queuing cars to start moving is measured. Then, the number of cars queuing in the lane when the traffic light turns red again is counted. The lane model is then tuned based on the collected data. The result of the tuning process is shown in Table 5. After the tuning the lane model becomes almost identical to the actual lane. This confirms the validity of the lane mode to be used in the case study.

2) Simulation Result. Furthermore, the model was used in a simulation with the duration of one hour. Both fixed-duration traffic light controller and fuzzy logic traffic light controller were tested under the same lane model and the data of *CI* was recorded from each observation.

The fuzzy logic traffic light controller is optimized using HS algorithm. The optimization parameters are chosen to be: $HMS = 5$, $HMCR = 0.7$, $PAR = 0.45$, and $Bw = 5$.

The optimized fuzzy logic traffic light controller is then tested to handle the other two unseen observation data. The results are compared with the result of the fixed-duration traffic light controller, as shown in Table 6.

VI. Conclusion

This paper discussed the optimization of fuzzy logic traffic light controller through its membership functions by using the Harmony Search Algorithm. The results show that the traffic light with optimized fuzzy logic controller can, by far, handle the traffic better than the fixed-duration traffic light controller. Moreover, the fuzzy logic controller with optimized membership function is designed to have additional features such as flexibility in determining which lane to be given green phase and for how long while at the same time avoiding the monopoly of green phase for by certain lane.

A case study was conducted on an intersection in Bekasi to provide a real data to test the controller, by first developing a lane model that can reflects the real traffic condition. The simulation shows that the optimized fuzzy logic controller outperforms the existing fixed-duration controller. Thus, the further implementation of the proposed fuzzy logic traffic light controller can be definitely endorsed in an attempt to resolve traffic problem.

TABLE V. COMPARISON BETWEEN OBSERVATION AND SIMULATION

Item	No.	Real	Model	Error
Initial QC (number of cars)	1	18	18	0
	2	39	39	0
	3	44	45	−1
	4	37	37	0
	5	40	40	0
Time needed to start moving (seconds)	1	16	15	1
	2	25	25	0
	3	29	29	0
	4	24	25	−1
	5	23	25	−2
Car stays when red again (number of cars)	1	8	7	1
	2	3	3	0
	3	2	4	−2
	4	5	5	0
	5	5	4	1

TABLE VI. COMPARISON OF AVERAGE WAITING TIME (IN SECONDS)

Controller	CI_{1S}	CI_{2S}	CI_{3S}
Optimized fuzzy logic	17.371	21.549	18.453
Fixed-duration	41.211	68.15	42.195
% of reduction	57.85%	68.38%	56.27%

References

[1] Tan, Kok Khiang, Khalid, M., Yusof, R., "Intelligent Traffic Lights Control by Fuzzy Logic," Malaysian Journal of Computer Science, Vol. 9 No. 2, pp. 29-35, 1996.

[2] Zarandi, M. H. F., Rezapour, Shabnam, "A Fuzzy Signal Controller for Isolated Intersections," Journal of Uncertain System, Vol. 3 No.3, pp. 174-182, 2009.

[3] Lee, K. S., and Z. W. Geem, "A New Meta-Heuristic Algorithm for Continuous Engineering Optimization: Harmony Search Theory and Practice," Computer Method in Applied Mechanics and Engineering, pp. 3902-393, 2005.

[4] Jantzen, Jan, "Foundation of Fuzzy Control," John Wiley & Sons, Ltd, 2007.

[5] Kovacic, Zdenko, and Bogdan, Stjepan, "Fuzzy Controller Design: Theory and Applications," Taylor & Francis Group, LLC, 2006.

[6] The MathWork, "Fuzzy Logic Toolbox User's Guide," The MathWork, Inc, 2002